

Universität Stuttgart

FACULTY 8 — INSTITUTE FOR THEORETICAL PHYSICS I
in cooperation with
FRAUNHOFER INSTITUTE FOR MANUFACTURING ENGINEERING AND
AUTOMATION (IPA)

Master's Thesis in Physics

Quantum Kernel Methods and Applications to Differential Equations

Author:	Roberto Flórez Abla
First Examiner:	Prof. Dr. Jörg Main
Second Examiner:	Prof. Dr. Mathias Scheurer
Supervisor:	Dr. Jan Schnabel (Fraunhofer IPA)
Submission Date:	October 16th, 2024

I confirm that this master's thesis is my own work and I have documented all sources and material used.

Roberto
Flórez
Ablan

Stuttgart, October 16th, 2024

Roberto Flórez Ablan

Abstract

Quantum computers have the potential to surpass classical computers in specific tasks, promising advantages in many fields. Machine Learning (ML), a domain with significant societal impact, is a key area of interest for exploring the applications of quantum computing. Here, we investigate two research directions aimed at understanding how current quantum computers can be used to solve ML problems.

First, we study Quantum Kernels (QKs). By calculating inner products between quantum states, QKs can be used to define similarity measures between points. QKs are a promising approach to Quantum Machine Learning (QML) but, in general, they have not been shown to outperform classical ML methods. A key reason for this is that QKs suffer from the exponential concentration problem. As the number of qubits increases, the kernel matrices become similar to the identity matrix, preventing generalization. One strategy to alleviate the exponential concentration problem is to rescale the data points that enter the quantum model. This technique is known as bandwidth tuning and has been shown to allow generalization in QKs. However, it has been numerically demonstrated that using this method results in QKs that cannot provide a quantum advantage over classical methods. In this thesis, we propose an explanation for this phenomenon. We show that due to the size of the rescaling factors, the QKs become similar to polynomial and RBF kernels, which are classically tractable.

Second, we implemented a Differential Equation (DE) solver based on variational quantum methods. A Quantum Neural Network (QNN) or QK, is used to represent an ansatz for the solution of a DE. The DE information is included into a loss function, which is minimized using a classical optimizer. In the case of a QK, the optimized parameters are the coefficients of a linear combination of QKs evaluated at the data points. In the case of a QNN, the optimized parameters are the phases of the quantum gates. The QNN implementation was included into the open-source QML python library sQUlearn. A preliminary hyperparameter study was conducted for QKs. Based on our limited investigation, we conclude that QKs leveraging the fidelity between quantum states, known as Fidelity Quantum Kernels (FQKs), demonstrate superior performance compared to those employing a semi-classical approach, referred to as Projected Quantum Kernels (PQKs).

Contents

Abstract	c
Abbreviations	g
1. Introduction	1
1.1. Motivation	1
1.2. Outline	2
2. Theory	4
2.1. The problem of supervised Machine Learning	4
2.2. Classical Machine Learning	5
2.2.1. Kernel Methods	5
2.2.2. Support Vector Machines (SVMs)	7
2.2.3. Neural Networks	9
2.2.4. Differential Equation Solvers using Parameterized Models	10
2.3. Variational Quantum Machine Learning	10
2.3.1. Quantum Kernel Methods	11
2.3.2. Quantum Neural Networks (QNNs)	14
2.3.3. Quantum Methods for Ordinary Differential Equations	15
3. Methods	19
3.1. Exponential Concentration	19
3.1.1. Sources of Exponential Concentration	21
3.2. Generalization on kernel methods	22
3.2.1. Spectral Bias and Task-Model Alignment	23
3.2.2. Maximum eigenvalue decay	26
3.2.3. Geometrical Difference	27
3.3. Bandwidth tuning	29
3.3.1. Generalization of bandwidth-tuned kernels	29
3.3.2. Numerical Evidence against bandwidth-tuned kernels	31
3.4. The complete picture	32

4. Results	34
4.1. Bandwidth-tuned Kernels are numerically similar to polynomial kernels	34
4.1.1. Motivating Example	35
4.1.2. Optimal polynomial kernels are very similar to optimal QKs for large number of dimensions	36
4.1.3. Models as a function of bandwidth	42
4.1.4. Analytical investigations of limiting cases of the variance of the FQKs	45
4.1.5. Comparing bandwidth-tuned quantum models with different classical models	47
4.1.6. Limitations	49
4.1.7. Discussion	51
4.2. Differential Equations Solvers using Variational Quantum Machine Learning	54
4.3. A small hyperparameter study of QKs as differential equation solvers .	54
4.3.1. Results	57
4.3.2. Discussion	59
5. Conclusion	61
List of Figures	63
List of Tables	71
Bibliography	72
A. Appendix	78
A.1. Matrix Inversion Lemma	78
A.2. Theory of a Reproducing Kernel Hilbert Space	78
A.3. Mercer’s Theorem	79
A.4. Technical details on numerical experiments of bandwidth-tuned kernels	80
A.5. Supporting figures for bandwidth-tuned kernels	82
A.6. Separable R_X encoding circuit	89
A.6.1. Analytical Variance of the Fidelity Quantum Kernel	90
A.7. Implementation of Quantum Differential Equation Solvers	93
A.7.1. QNN Differential Equation Solver	94
A.7.2. QK Differential Equation Solver	94
A.8. Gradient of the loss of a DE	97
A.9. Derivatives of RBF PQKs	97
A.9.1. First derivative	97

Contents

A.9.2. Second derivative	98
Acknowledgments	99

Abbreviations

QML Quantum Machine Learning

QNN Quantum Neural Network

QK Quantum Kernel

FQK Fidelity Quantum Kernel

PQK Projected Quantum Kernel

RKHS Reproducing Kernel Hilbert Space

SVM Support Vector Machine

ML Machine Learning

NN Neural Network

DE Differential Equation

RBF Radial Basis Function

KRR Kernel Ridge Regression

SVC Support Vector Classification

MSE Mean Squared Error

VQA Variational Quantum Algorithm

PDE Partial Differential Equation

ODE Ordinary Differential Equation

IQP Instantaneous Quantum Polynomial

roc auc Receiver Operating Characteristic Area Under the Curve

PCA Principal Component Analysis

1. Introduction

1.1. Motivation

Shor’s algorithm [1] demonstrated that a quantum computer can factorize large numbers in polynomial time, whereas the fastest known classical method requires exponential time [2]. Additionally, significant experimental progress has been made in the manipulation of individual quantum systems [3–5]. Motivated by these discoveries, the field of quantum computing has experienced rapid growth over the past decade. One recent research direction is to understand how (or if) quantum computers can be used to solve ML problems [6–8]. This new field is known as Quantum Machine Learning (QML). Within QML, a largely studied approach is the use of variational quantum circuits [6, 7, 9–11]. Fundamentally, information is encoded into a quantum computer by preparing a quantum state that can be manipulated using quantum gates. The state is then measured, and the outcome is used to make predictions. The two most popular approaches within this framework are QNNs and QKs. QNNs are quantum circuits that are trained to perform specific tasks, while QKs use quantum circuits to define similarity measurements between data points in a quantum mechanical way.

This thesis is divided into two complementary research directions:

1. We explore the role of the bandwidth tuning technique [12–14] for quantum kernels. This technique consists of rescaling the size of data points that enter the quantum model, and it has been shown to improve the performance of quantum kernels in classification tasks. At the same time, it was numerically demonstrated that using this method results in classically tractable QKs [14]. However, why bandwidth-tuned kernels become classically tractable was not clear. In this thesis, we propose an explanation for this phenomenon. Sufficiently small bandwidths, scale down drastically quantum kernel such that the models are comparable to polynomially tractable functions.
2. We implement and incorporate a differential equation solver based on variational quantum methods [15, 16] to the Python library sQUlearn [17]. The QNNs methods and the key ingredient for the QKs are already available. Although

much research exist on differential equation solvers using variational quantum methods [15, 16, 18–20], to the best of our knowledge, no public library has implemented these methods for both QNNs and QK.

1.2. Outline

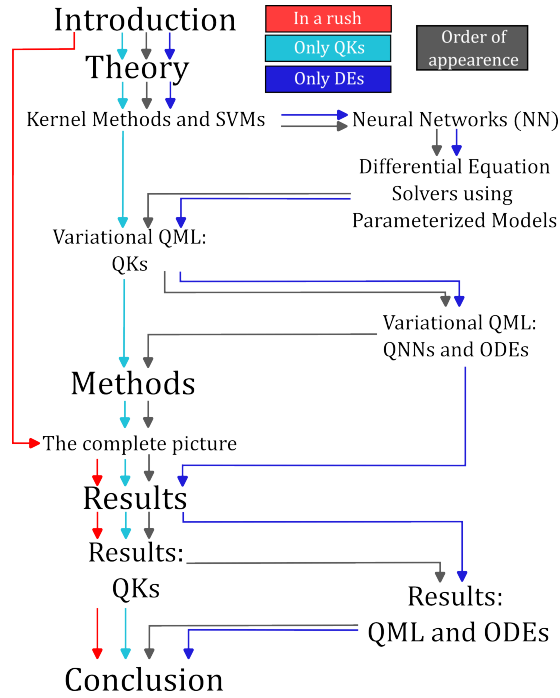


Figure 1.1.: Outline of the thesis, four reading paths are suggested. In-between sections are omitted.

The overall structure of the thesis is as follows.

In Chapter 2, we introduce some basic concepts of ML and QML, we also discuss the fundamentals to solve differential equations using variational methods. In Chapter 3, we explain the state of the art of quantum kernels. In Chapter 4, Section 4.1 we present the results of our research on bandwidth-tuned kernels and in Section 4.2 and Section 4.3 we present the results of our research on differential equation solvers. Finally, in Chapter 5, we present a summary of the thesis.

The manuscript was written to be accessible on a linear reading. Nevertheless, alternative reading paths are possible (see Fig. 1.1). If the reader is in a rush or is familiar

with the concepts of ML and/or QML (see red path), we recommend reading Section 3.4 and then move on to the results of Chapter 4. If the reader is only interested in a more complete look at the quantum kernels, we refer to the light blue path of Fig. 1.1. Similarly, for interest only on the differential equation solvers, we recommend the blue path.

2. Theory

2.1. The problem of supervised Machine Learning

Supervised ML consists on learning a function that maps inputs to outputs by looking at examples. Given the observations $\{(x_1, y_1), \dots, (x_N, y_N)\}$, the goal is to find a function f_θ that resembles the true observation y_i , for unseen data, as closely as possible. Fundamentally, the learning problem of supervised ML can be understood by the concept of Empirical Risk Minimization [21], which is formalized by the following equation,

$$\min_{f_\theta \in \mathcal{F}} \frac{1}{N} \sum_{i=1}^N \mathcal{L}(f_\theta(x_i), y_i). \quad (2.1)$$

We aim at finding the optimal parameters θ_{opt} that minimize a loss function $\mathcal{L}$, given a set of functions $f_\theta \in \mathcal{F}$ which are available. The loss function $\mathcal{L}$ compares the prediction $f_\theta(x_i)$ with regards to the true values y_i . To solve this fundamental equation, we need to make four main design choices [21] that cover many of the topics in supervised ML [22, 23]. These are,

1. The hypothesis space $\mathcal{F}$, which is the set of functions that we consider to be the solution to the learning problem. For example, Neural Networks (NNs), Support Vector Machines (SVMs), Decision Trees, etc.
2. The loss function $\mathcal{L}$, which measures the quality of the prediction. For example, the Mean Squared Error (MSE), the Mean Absolute Error, Cross-Entropy, etc.
3. The optimization algorithm, which finds the optimal parameters θ_{opt} . For example, Gradient Descent, Adam, etc.
4. The regularization technique, which prevents overfitting. For example, L_1 , L_2 regularization, Dropout, etc.

Points 2 to 4 are shared by ML and QML. The main difference between the two fields is the hypothesis space $\mathcal{F}$. In QML, the hypothesis space is given by functions evaluated by a quantum computer. In this chapter we present the main theoretical background to

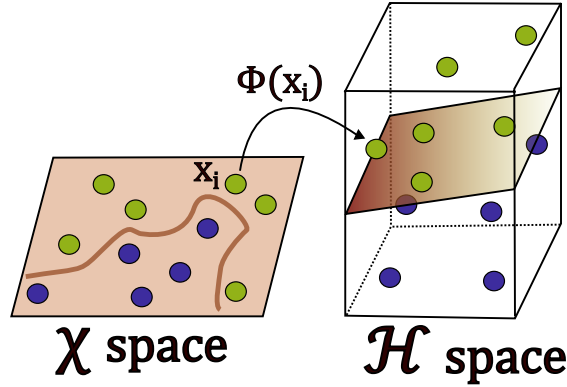


Figure 2.1.: A Classification problem in a two-dimensional space is transformed to a three-dimensional space using a kernel function. The decision boundary becomes linear in the higher-dimensional space.

understand the tools used in this work. We introduce the classical methods in Section 2.2, and their quantum counterparts in Section 2.3.

2.2. Classical Machine Learning

2.2.1. Kernel Methods

Kernels are a very successful method in Machine Learning [21, 22, 24]. Fundamentally, they map points to a higher-dimensional space (potentially infinite) where the learning problem can become linear (Fig. 2.1). The high-dimensional space is accessed by calculating inner products, instead of computationally costly explicit mapping. By defining an inner product, kernels act as a distance metric or a similarity measure between points in this higher dimensional space.

To introduce kernels as a method, let us tackle the problem of regression (predicting a continuous variable), formally called Kernel Ridge Regression (KRR). The main ideas will carry over to classification problems.

Consider a training dataset $\mathcal{D}$ of N pairs $\{\{x_i, y_i\}\}^N$, where $x_i \in \mathcal{X} = \mathbb{R}^d$ and $y_i \in \mathbb{R}$. We assume that the y_i are produced by an unknown target function $\bar{f}(x_i)$ that we wish to learn. Each point is sampled from an unknown distribution $p(x, y)$. To solve the regression problem, we assume an ansatz function given by,

$$f(x) = \sum_{j=1}^p w_j \phi(x)_j = \mathbf{w}^T \boldsymbol{\phi}(x), \quad (2.2)$$

where $\phi(x) \in \mathcal{H}$ represents a **feature map**. A feature map is a function that maps the input $x \in \mathcal{X}$ to $\mathcal{H}$, a dot product space. For simplicity, take $\mathcal{H} = \mathbb{R}^{1 \times p}$, (generally, this is not necessary, and it is discussed on Appendix A.2). Also, $w \in \mathbb{R}^{p \times 1}$ are the weights that we aim to learn. To find the ideal ansatz solution we define a loss function $\mathcal{L} = \mathcal{L}(w)$, which measures how well the prediction $f(x_i)$ resembles the true values, y_i . $\mathcal{L}(w)$ can be optimized, and the optimal weights can be found. We will see that assuming Eq. 2.2, and minimizing a loss function given by the MSE, the concept of kernels,

$$k(x, x') = \phi(x)\phi(x')^T, \quad (2.3)$$

emerges naturally.

The MSE, between our training data and the ansatz function, is given by,

$$\mathcal{L}(w) = \frac{1}{N} \sum_{i=1}^N (y_i - f(x_i))^2 + \lambda \|w\|^2 \quad (2.4)$$

$$= \frac{1}{N} \sum_{i=1}^N (y_i - w^T \phi(x_i))^2 + \lambda \|w\|^2, \quad (2.5)$$

$$= \frac{1}{N} (Y - w^T \Phi)^T (Y - \Phi w) + \lambda w^T w, \quad (2.6)$$

$$= \frac{1}{N} (Y^T Y - 2Y^T \Phi w + w^T \Phi^T \Phi w + \lambda w^T w), \quad (2.7)$$

where λ is a regularization parameter¹. The regularization term is added to prevent singular matrices and to avoid ill-conditioning of the optimization problem. From Eq. 2.5 to Eq. 2.6, we have defined the matrices $Y = [y_1, \dots, y_N]^T \in \mathbb{R}^{N \times 1}$ and $\Phi = [\phi(x_1), \dots, \phi(x_N)]^T \in \mathbb{R}^{N \times p}$. To find the weights that minimize $\mathcal{L}$, we take the derivative with respect to w and set it to zero,

$$\frac{\partial \mathcal{L}(w)}{\partial w} = \frac{1}{N} (-2Y^T \Phi + 2w^T \Phi^T \Phi + 2\lambda w^T) = 0 \in \mathbb{R}^{1 \times p} \implies \quad (2.8)$$

$$w^T = (Y^T \Phi) (\Phi^T \Phi + \lambda \mathbb{I}_p)^{-1} \iff \quad (2.9)$$

$$w = (\Phi^T \Phi + \lambda \mathbb{I}_p)^{-1} \Phi^T Y. \quad (2.10)$$

Eq. 2.10 can be rewritten using the Matrix Inversion Lemma [22] (see Appendix A.1),

¹Also called Ridge, Tikhonov or L_2 regularization

and the optimal weights are given by,

$$\mathbf{w} = (\Phi^T \Phi + \lambda \mathbb{I}_p)^{-1} Y^T \Phi = \Phi^T (\Phi \Phi^T + \lambda \mathbb{I}_N)^{-1} Y \implies \quad (2.11)$$

$$\mathbf{w}^T = Y^T (\Phi \Phi^T + \lambda \mathbb{I}_N)^{-1} \Phi \quad (2.12)$$

$$= \sum_i \alpha_i \phi(x_i)^T, \quad \alpha = (\Phi \Phi^T + \lambda \mathbb{I}_N)^{-1} Y. \quad (2.13)$$

Where we have introduced the coefficients α_i that depend on the inner products of feature maps. Thus, the prediction for a new point x is,

$$f(x) = \mathbf{w}^T \phi(x) \quad (2.14)$$

$$= Y^T (\Phi \Phi^T + \lambda \mathbb{I}_N)^{-1} \Phi \phi(x) \quad (2.15)$$

$$= \sum_i \alpha_i \phi(x_i)^T \phi(x) \quad (2.16)$$

$$= \sum_i \alpha_i k(x_i, x), \quad (2.17)$$

Some relevant points regarding this derivation.

- The prediction of the model $f(x)$ can be written in terms of linear combination of inner products. These are given by the kernel function evaluated at the input points. Thus, calculating explicitly $\phi(x)$ is not necessary. This is known as the **kernel trick**. The formal result is called the **Representer Theorem** [24].
- Due to the previous point, infinite-dimensional feature maps can be used. For example, see the Gaussian Radial Basis Function (RBF) Kernel, in Table 2.1.
- As the coefficients α_i are given by an analytical expression, this is a convex optimization problem and a guaranteed solution to the optimization problem exists.
- For k to be a valid kernel function, the matrix formed by evaluating the pairwise kernel functions, must be symmetric and positive semi-definite [22, 24] (see Appendix A.2).

Once we have introduced the concept of kernels, we can introduce SVMs, another fundamental method in ML that uses kernels.

2.2.2. Support Vector Machines (SVMs)

SVMs or Support Vector Classifications (SVCs) are a kernel algorithm used for classification, where an optimally separating hyperplane is found. The hyperplane is

Table 2.1.: Some examples of kernel functions and their corresponding feature maps, for one-dimensional variables $x, y \in \mathbb{R}^1$.

Kernel	$K(x, y)$	$\phi(x)$
Polynomial Kernel $d = 1$	$(x^T y + \gamma)^1$	$[\sqrt{\gamma}, x]$
Polynomial Kernel $d = 2$	$(x^T y + \gamma)^2$	$[\gamma, \sqrt{2\gamma}x, x^2]$
Polynomial Kernel $d = 3$	$(x^T y + \gamma)^3$	$[\gamma, \sqrt{3\gamma^2}x, \sqrt{3\gamma}x^2, x^3]$
Gaussian RBF Kernel	$\exp(-\gamma\ x - y\ ^2)$	$\exp(-\gamma x^2)[1, \sqrt{\frac{2\gamma}{1!}}x, \sqrt{\frac{(2\gamma)^2}{2!}}x^2, \dots]$

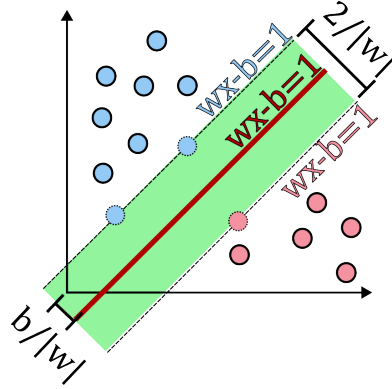


Figure 2.2.: Schematic representation of the largest margin $2/\|w\|$ separating hyperplane in a two-dimensional space. The closest points to the hyperplane, with dotted lines, are the support vectors. Based on Ref. [25].

characterized by the support vectors, which are the points closest to the separating surface (see Fig. 2.2). The optimization problem consists on maximizing the margin, $2/\|w\|$, that corresponds to the distance between the separating hyperplane and the support vectors, such that all points are correctly separated. The optimization problem can be written as,

$$\max_{\alpha} \min_{w, b} L(w, b, \alpha) = \max_{\alpha} \min_{w, b} \frac{1}{2} \|w\|^2 - \sum_{i=1}^N \alpha_i (y_i (w^T \phi(x_i) + b) - 1), \quad (2.18)$$

$$\text{subject to } \alpha_i \geq 0, \quad \forall i. \quad (2.19)$$

This is known as the dual problem. By optimizing the Lagrangian above, a solution can be found to be a convex quadratic optimization problem [24], given by,

$$\max_{\alpha} \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \phi(x_i)^T \phi(x_j), \quad \text{subject to } \sum_{i=1}^N \alpha_i y_i = 0, \quad \alpha_i \geq 0, \quad \forall i. \quad (2.20)$$

Once again, the solution only depends on the inner products of the feature maps. Thus, the definition of the kernel function, Eq. 2.3, can be used. Later, we will see that the inner products used to define kernels in SVMs or KRR can be calculated by a quantum computer.

Another technique which has been very popular in ML and has a quantum counterpart are Neural Networks (NNs).

2.2.3. Neural Networks

NNs are a very successful ML technique [21, 23]. They consist of composed nonlinear functions, where each nonlinear function is known as a layer. The most common architecture is the feedforward NN, where the output of one layer is the input of the next layer. A layer i is given by,

$$h^{(i)}(x) = \sigma^{(i)}(W^{(i)}x + b^{(i)}), \quad (2.21)$$

where $h^{(i)}$ is the output of the layer, $W^{(i)}$ is a weight matrix, $b^{(i)}$ is a bias vector, and $\sigma^{(i)}$ is a nonlinear activation function. Weights and biases are completely parameterized matrices of the NN that are learned during training. On the other hand, activation functions are decided by design, examples of the type of functions can be seen in Table 2.2. For an L -layer NN, the output is given by,

$$f(x) = h^{(L)}\left(h^{(L-1)}\left(\dots\left(h^{(1)}(x)\right)\right)\right). \quad (2.22)$$

Table 2.2.: Activation Functions for Neural Networks.

Name	Activation Function
Sigmoid	$\sigma(x) = \frac{1}{1+\exp(-x)}$
Tanh	$\sigma(x) = \tanh(x)$
ReLU	$\sigma(x) = \max(0, x)$
Leaky ReLU	$\sigma(x) = \max(ax, x)$

The supervised ML techniques discussed earlier have been extended to problems beyond regression and classification. For instance, NNs have been used to solve Partial Differential Equations (PDEs) and Ordinary Differential Equations (ODEs). In the following subsection, we will introduce the main ideas for variationally solving DEs.

2.2.4. Differential Equation Solvers using Parameterized Models

DEs are building blocks of many scientific fields [26]. Numerical Methods to solve DEs are split in two [15]: local and global methods. Local methods (i.e Runge-Kutta and finite differences) approximate differentials by discrete sums and recursively find the numerical solution. On the other hand, global methods represent an ansatz in a suitable basis and solve for the optimal coefficients of the numerical differential equation in this basis. Global methods have been used extensively in classical Machine Learning. One known strategy is called Physical-Informed Neural Networks [27, 28]. In this case, the ansatz is given by a NN and the information about the derivatives is included into the loss function to be optimized. Let us formally define this idea, consider a DE given in its functional form,

$$F = F[\{d^m f_n/dx^m\}_{m,n}, \{f_n(x)\}_n, x] = 0, \quad (2.23)$$

where f corresponds to the function solution of the DE. f can be represented in a parameterized differentiable form,

$$f = f_\theta, \quad (2.24)$$

and the solution to the DE can be seen as an optimization problem. A loss function,

$$\mathcal{L}_\theta = \mathcal{L}_\theta(\{d^m f_n/dx^m\}_{m,n}, \{f_n(x)\}_n, x), \quad (2.25)$$

is optimized to find the ideal parameters θ_{opt} .

$$\theta_{\text{opt}} = \arg_\theta \min(\mathcal{L}_\theta[d_x f, f, x]). \quad (2.26)$$

In Section 2.3.3, we will further comment on this idea by replacing the function ansatz $f(x)$ with differentiable quantum circuits based on QKs and QNNs.

In the next section, we will define and replace the quantum counterparts to the classical ML methods that have been introduced.

2.3. Variational Quantum Machine Learning

Current quantum computers are noisy and have a limited number of qubits, making fault-tolerant quantum algorithms not possible yet [9]. As an alternative, Variational Quantum Algorithms (VQAs) have been proposed to solve ML problems with current hardware [9]. Two very popular approaches for variational QML are QNNs [20, 29–32] and QKs [6, 33–37]. In this section, we will introduce the main ideas behind these methods.

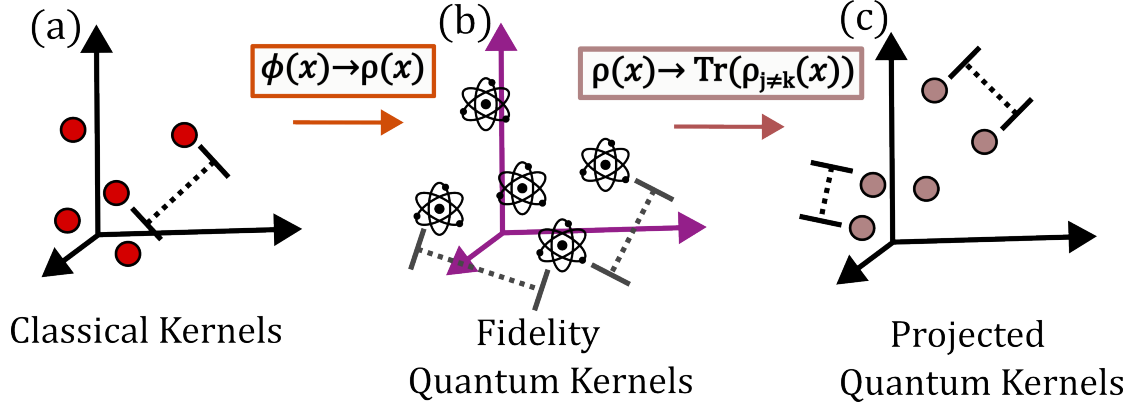


Figure 2.3.: A visual representation of the difference between classical and QKs. In a FQK, points are mapped from a classical space to a quantum space through a density matrix, and the inner product corresponds to the fidelity between the two encoded states. PQKs represent an in-between approach, where the quantum state is projected to a classical space and is classically manipulated by an envelope function. Inspired by Fig. 2 from Ref. [10]

2.3.1. Quantum Kernel Methods

Classical Kernels define a similarity measure between two points and then use this measure to make predictions by linearly combining the similarities. QKs follow the same idea, but the similarities are defined by measurements in a quantum computer. The most popular approaches to QKs are Fidelity Quantum Kernels (FQKs) [11] and Projected Quantum Kernels (PQKs) [10]. In this section, we will formally define these two methods. Before that, let us introduce the concept of quantum computations, the operations that are performed in a quantum computer.

Definition 1. Quantum computations [2, 36]. Consider a quantum computer with n qubits. Let $\mathcal{H}$ be the space of density matrices $\rho \in \mathbb{C}^{2^n \times 2^n}$, equipped with the Hilbert-Schmidt inner product $\langle \rho, \rho' \rangle_{\mathcal{H}} = \text{Tr}(\rho^\dagger \rho')$. Let $\rho = |\psi\rangle\langle\psi|$ be the initial state of the quantum mechanical system. A quantum computer implements the following mapping,

$$\rho \rightarrow U\rho U^\dagger = \rho', \quad (2.27)$$

where $U = U^\dagger$ is a unitary operator.

Quantum computations can encode data to a quantum state by using data-dependent unitaries. With this data-encoding circuits or unitaries, we can define the quantum counterparts to classical kernels.

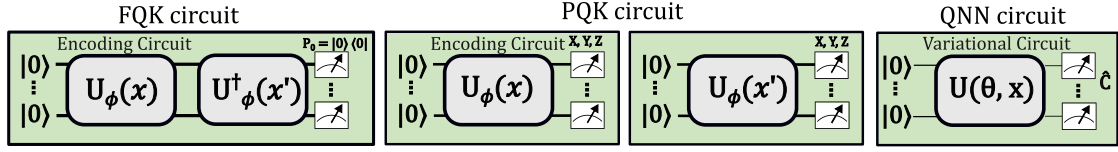


Figure 2.4.: Schematic representation of circuits used to compute the most popular approaches in QML. FQKs and PQKs compare two points, while QNNs are a general ansatz for a function. Inspired by Fig. 1 from Ref. [37]

Fidelity Quantum Kernels (FQKs)

Following Ref. [36], we will define the data-encoding feature map that gives rise to FQKs.

Definition 2. Data-encoding feature map [36]. *Let $\mathcal{X}$ be a nonempty set containing the input data. A Data-encoding feature map is a function $\phi : \mathcal{X} \rightarrow \mathcal{H}$ that maps data $x \in \mathcal{X}$ to a density matrix $\rho \in \mathcal{H}$ by a data-encoding unitary operator $U(x)$ acting on a state ρ_0 ,*

$$\phi(x) = \rho(x) = |\phi(x)\rangle \langle \phi(x)| = U(x)\rho_0 U(x)^\dagger. \quad (2.28)$$

Following our previous discussion, a kernel can be defined by the inner product in the mapped feature space. In this case, the inner-product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ of the mapped feature space $\mathcal{H}$, is the Hilbert-Schmidt inner product $\text{Tr}(\cdot, \cdot)$. Thus, we arrive at the definition of a FQK.

Definition 3. Fidelity Quantum Kernel [10, 11, 36]. *Let $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$, be a function that compares the fidelity between two density matrices $\rho(x)$ and $\rho(x')$, given by,*

$$k(x, x') = \langle \phi(x), \phi(x') \rangle_{\mathcal{H}} = \text{Tr}(\rho(x)\rho(x')) = |\langle \phi(x) | \phi(x') \rangle|^2. \quad (2.29)$$

In Fig. 2.3, we show a cartoon representation of a mapping from a classical space to a density matrix, where then similarities are measured by the fidelity between two states. One way to compute Eq. 2.29 in a quantum computer, is by implementing the quantum circuit² shown in Fig. 2.4. To see this, consider the initial state $|\psi\rangle = |0\rangle$, applying the sequence of unitaries shown in the figure, we obtain $|\psi\rangle = U_\phi^\dagger(x')U_\phi(x)|0\rangle$ and measuring the resulting state $|\psi\rangle$ against the $\hat{P}_0 = |0\rangle\langle 0|$ operator. The measurements

²An alternative way to calculate Eq. 2.29 is by the so-called swap test [7]

results in,

$$\langle \hat{P}_0 \rangle = \text{Tr}(|\psi(x)\rangle \langle \psi(x)| \hat{P}_0) \quad (2.30)$$

$$= \langle \psi(x) | \hat{P}_0 | \psi(x) \rangle \quad (2.31)$$

$$= \langle 0 | U_\phi^\dagger(x') U_\phi(x) | 0 \rangle \langle 0 | U_\phi^\dagger(x') U_\phi(x) | 0 \rangle \quad (2.32)$$

$$= \langle \psi(x') | \psi(x) \rangle \langle \psi(x') | \psi(x) \rangle = |\langle \phi(x) | \phi(x') \rangle|^2. \quad (2.33)$$

Example of FQK

A simple analytically tractable example of a FQK can be built by considering only rotations around the x axis in the Bloch sphere. This example will be later used in Section 3.2 and Section 4.1.4 as a toy problem³. Let us apply a rotation $R_x(x)$ to the state $|0\rangle$,

$$|\psi(x)\rangle = R_x(x) |0\rangle = \cos\left(\frac{x}{2}\right) |0\rangle + \sin\left(\frac{x}{2}\right) |1\rangle, \quad (2.34)$$

calculating the fidelity between two density matrices after applying the rotations, we get,

$$k(x, x') = \text{Tr}(\rho(x)\rho(x')) = \cos^2\left(\frac{(x - x')}{2}\right) \quad (2.35)$$

For n qubits with a corresponding rotation,

$$k(\mathbf{x}, \mathbf{x}') = \prod_{i=1}^n \cos^2\left(\frac{(x_i - x'_i)}{2}\right). \quad (2.36)$$

Note, it is no coincidence that the final kernel is a product of trigonometric functions. As a matter of fact, this turns out to be a general property of FQKs, they can be written as a partial Fourier series (see Section 4B of Ref. [36]),

$$k(\mathbf{x}, \mathbf{x}') = \sum_{\mathbf{s}, \mathbf{t} \in \Omega} e^{i\mathbf{s}\mathbf{x}} e^{i\mathbf{t}\mathbf{x}'} c_{\mathbf{s}\mathbf{t}}, \quad (2.37)$$

where the frequencies Ω are determined by the generating Hamiltonian of the quantum circuit and $c_{\mathbf{s}\mathbf{t}}$ are the Fourier coefficients.

Strong empirical evidence of numerical advantage of FQK has not been found until today, in most cases, FQK are inferior or just competitive with classical kernels [10, 14, 38]. Nevertheless, in a specific artificial classification dataset based on the discrete logarithm problem, Liu *et al.* [39] rigorously proved that FQKs can provide an exponential

³A more general derivation is available in Appendix A.6

advantage in generalization to classical methods. In this work, Shor’s algorithm was used as a subroutine.

One motivation for FQKs is that they have access to an exponentially (with the number of qubits) large Hilbert space, which could provide some type of advantage [9, 40, 41]. At the same time, calculating inner products in a very large space can make learning algorithms inefficient [10, 42–44] (see Section 3.1). As an alternative, Ref. [10] proposed PQKs.

Projected Quantum Kernels

Instead of using the exponentially large quantum Hilbert space, a PQK projects the quantum state to a lower-dimensional space, where quantities can be classically manipulated (see Fig. 2.3). One form to do this is to consider one-particle reduced density matrices (1 – RDM), $\rho_k(x) = \text{Tr}_{j \neq k}(\rho(x))$. Therefore, the PQK is defined by,

Definition 4. Projected Quantum Kernel [10]. *Let $\rho_k(x)$ be the 1 – RDM of the quantum state $\rho(x)$, and $g(\rho_k(x), \rho_k(x'))$ be an arbitrary envelope function, then the PQK is defined by,*

$$k(x, x') = g(\rho_k(x), \rho_k(x')). \quad (2.38)$$

For example, the RBF-like PQK is given by,

$$k(x, x') = \exp \left(-\gamma \sum_{k, P \in X, Y, Z} (\text{Tr}(P\rho_k(x)) - \text{Tr}(P\rho_k(x')))^2 \right), \quad (2.39)$$

where the expectation values can be calculated by expectation over local observables $\text{Tr}(P\rho_k(x')) = \langle P_k \rangle$. Instead of comparing two points in a quantum Hilbert space by using kernels, we can directly manipulate and encode information on a quantum computer to solve ML problems. This is the idea behind QNNs.

2.3.2. Quantum Neural Networks (QNNs)

QNNs follow similar principles to classical NNs, but each layer originating a weight matrix now corresponds to parameterized unitary operators. Similar to FQKs, it has been shown that QNNs can be written as partial Fourier series [20, 36, 45], where the frequencies are determined by the generating Hamiltonian of a quantum circuit.

Example of a QNN

A simple example of a QNN can be built by considering a quantum circuit that applies a rotation $R_y(p)R_x(x)$ to the state $|0\rangle$, where p is a parameter, measuring the expectation value against Z , we get,

$$f(x) = \langle 0 | R_x(x)^\dagger R_y(p)^\dagger Z R_y(p) R_x(x) | 0 \rangle = \cos(x) \cos(p). \quad (2.40)$$

This set of methods can be used for regression and classification problems, but they have also been suggested to solve DEs [15, 16].

2.3.3. Quantum Methods for Ordinary Differential Equations

In Section 2.2.4, we saw that DE can be solved by differentiable parameterized ansätze by minimizing a loss function. When using quantum models given by parameterized circuits, differentiation can be done analytically by the parameter-shift rule [46–49].

Parameter-Shift Rule

The parameter-shift rule is a method to exactly calculate the gradient of a quantum circuit by resifting the arguments used in the circuit. To motivate the parameter-shift rule, consider

$$f(x) = \sin(x), \quad (2.41)$$

then the derivative is given by,

$$\frac{df}{dx}(x) = \cos(x) = \frac{\sin(x + \pi/2)}{2} + \frac{\sin(x - \pi/2)}{2}, \quad (2.42)$$

where we used the trigonometric identity $\sin(a + b) = \sin(a) \cos(b) + \cos(a) \sin(b)$. The final derivative, only depends on the initial function but with shifted arguments. The parameter-shift rule generalizes this idea to quantum circuits:

Definition 5. Parameter-Shift Rule [46–49]. Let $U_G(\theta) = e^{-ia\theta G}$ be a parameterized gate of parameter θ , real constant a and Hermitian generator G , with two unique eigenvalues e_0 and e_1 . For an objective function, given by the expectation value of an observable A ,

$$f(\theta) = \langle \psi | U_G(\theta) A U_G(\theta)^\dagger | \psi \rangle, \quad (2.43)$$

the derivative of f with respect to θ is given by,

$$\frac{df(\theta)}{d\theta} = r \left(f\left(\theta + \frac{\pi}{2r}\right) - f\left(\theta - \frac{\pi}{2r}\right) \right), \quad (2.44)$$

with $r = \frac{a}{2} (e_1 - e_0)$.

2. Theory

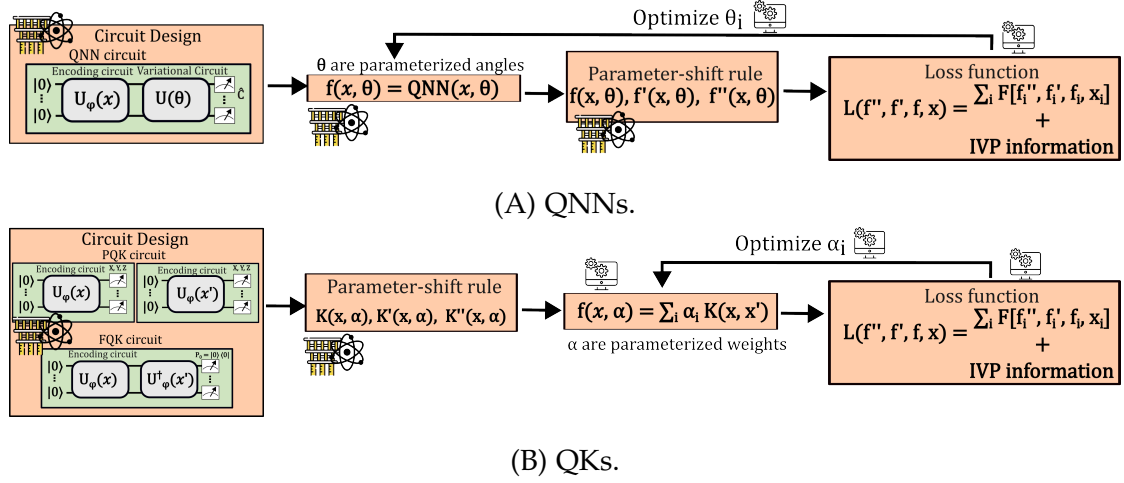


Figure 2.5.: Scheme representing the optimization process to solve a DE using a QNN (A) and a QK (B). The quantum circuit is parameterized by a QNN or a QK. The functions are evaluated on a quantum computer and optimized classically. Inspired by Fig. 1 from Ref. [15].

The differentiation can be extended to more general circuits, gradients and higher-order derivatives⁴ [46]. By exploiting the differentiable properties of parameterized quantum circuits, Kyriienko *et al.* proposed to construct a differential equation loss with trial functions given by parameterized quantum circuits that can be optimized classically. The trial function are given by QNNs [15] or QKs [16].

Quantum Neural Network Methods

A QNN is set up to encode a trial function that will be the solution of the DE. Using an encoding unitary $U_\phi(x)$, and a parameterized set of gates U_θ , a trial state $|f_{\theta,\phi}(x)\rangle = U_\theta U_\phi(x) |0\rangle$ is constructed. By measuring this state with regard to an arbitrary cost operator $\hat{C}$, the final trial function is given by,

$$f_{\theta,\phi}(x) = \langle \hat{C} \rangle = \langle f_{\theta,\phi}(x) | \hat{C} | f_{\theta,\phi}(x) \rangle. \quad (2.45)$$

ϕ is a nonlinear function that is used to preprocess the data before encoding it into the quantum state. The encoded state after the first unitary $U_\phi(x)$ is then manipulated by the parameterized unitary U_θ .

⁴In practice, the parameter-shift rule is already implemented in general quantum programming libraries, such as PennyLane [50], Qiskit [51] and sQULearn [17]

The derivatives of this ansatz can be calculated using the parameter-shift rule (Eq. 2.44). Using the derivatives, a loss function as defined in Section 2.2.4 can be built,

$$\mathcal{L}(\theta) = \sum_{i=1}^N \left(F \left[x_i, f_{\theta, \phi}(x_i), \frac{df_{\theta, \phi}(x_i)}{dx_i} \right] \right)^2, \quad (2.46)$$

and classically optimized. The workflow of this method is shown in Fig. 2.5A.

Initial Boundary condition

Information about the initial conditions of the DE must be included in the loss function. Ref. [15] proposes three methods,

1. **Pinned Boundary Condition.** An extra term $\mathcal{L}_{IV}$ is added to the loss function $\mathcal{L}_f$ to enforce the initial values of the DE. This term is "pinned" by a constant parameter η . The loss function is $\mathcal{L}_{total}$ given by:

$$\mathcal{L}_{total}(\theta) = \mathcal{L}_f + \eta \mathcal{L}_{IV} \quad (2.47)$$

$$= \sum_{i=1}^N \left(F \left[x_i, f_{\theta, \phi}(x_i), \frac{df_{\theta, \phi}(x_i)}{dx_i} \right] \right)^2 + \eta (f_{\theta, \phi}(x_0) - u_0)^2. \quad (2.48)$$

2. **Floating Boundary Condition.** A bias term $f_b(\theta, u_0) = u_0 - \langle f_{\theta, \phi}(x_0) | \hat{C} | f_{\theta, \phi}(x_0) \rangle$ is included in the trial function, such that the new trial function "floats" around the initial condition. The loss function is given by:

$$\mathcal{L}_{total}(\theta) = \sum_{i=0}^N \left(F \left[x_i, f_{\theta, \phi}(x_i), \frac{df_{\theta, \phi}(x_i)}{dx_i} \right] \right)^2 \quad (2.49)$$

$$\text{with } f_{\theta, \phi}(x) = f_b(\theta, u_0) + \langle f_{\theta, \phi}(x) | \hat{C} | f_{\theta, \phi}(x) \rangle. \quad (2.50)$$

3. **Optimized Boundary Condition.** Similar to the previous one, but the bias term is a fully classical parameter that does not directly include the IVP information, $f_c \in \mathbb{R}$. The loss function is given by:

$$\mathcal{L}_{total}(\theta) = \sum_{i=0}^N \left(F \left[x_i, f_{\theta, \phi}(x_i), \frac{df_{\theta, \phi}(x_i)}{dx_i} \right] \right)^2 \quad (2.51)$$

$$\text{with } f_{\theta, \phi}(x) = f_c + \langle f_{\theta, \phi}(x) | \hat{C} | f_{\theta, \phi}(x) \rangle. \quad (2.52)$$

QNNs solve DEs by constructing a trial function using quantum circuits. As already mentioned, another way to solve DEs is by using QKs which encode the information in the higher-dimensional Hilbert space defined by the inner products.

Kernel Methods

To solve DEs using (quantum or classical) kernels, inspired by the representer theorem, a function ansatz is given by a linear combination of kernel inner products,

$$f_{\alpha}(x) = \sum_{i=1}^N \alpha_i k(x, y_i), \quad (2.53)$$

where the coefficients α are found by minimizing a loss function. This is different from Eq. 2.17, where the coefficients were given by the representer theorem. In this case, the y_i correspond to the numerical domain of the DE. If the kernel function is differentiable, the derivative is given by,

$$\frac{df_{\alpha}(x)}{dx} = \sum_{i=1}^N \alpha_i \frac{\partial k(x, y_i)}{\partial x}. \quad (2.54)$$

Using the model constructed from the kernel (Eq. 2.53 and Eq. 2.54), the loss function can be built by substituting the derivatives in the DE,

$$\mathcal{L}(\alpha) = \sum_{i=1}^N \left(F \left[x_i, f_{\alpha}(x_i), \frac{df_{\alpha}(x_i)}{dx_i} \right] \right)^2. \quad (2.55)$$

For kernels, the boundary conditions mentioned in Section 2.3.3 also apply. After the loss function is defined, the optimization is done classically. The workflow of this method is shown in Fig. 2.5B. One advantage of using kernels, is that the loss function only depends on the classical parameters α and the choice of kernel. Therefore, once a kernel is chosen and evaluated at the numerical domain y_i , the optimization does not require using again the quantum computer. The minimization problem can be a convex optimization if $\frac{\partial^2 \mathcal{L}}{\partial \alpha_j^2} \geq 0 \forall \alpha_j$, thus a global minimum can exist.

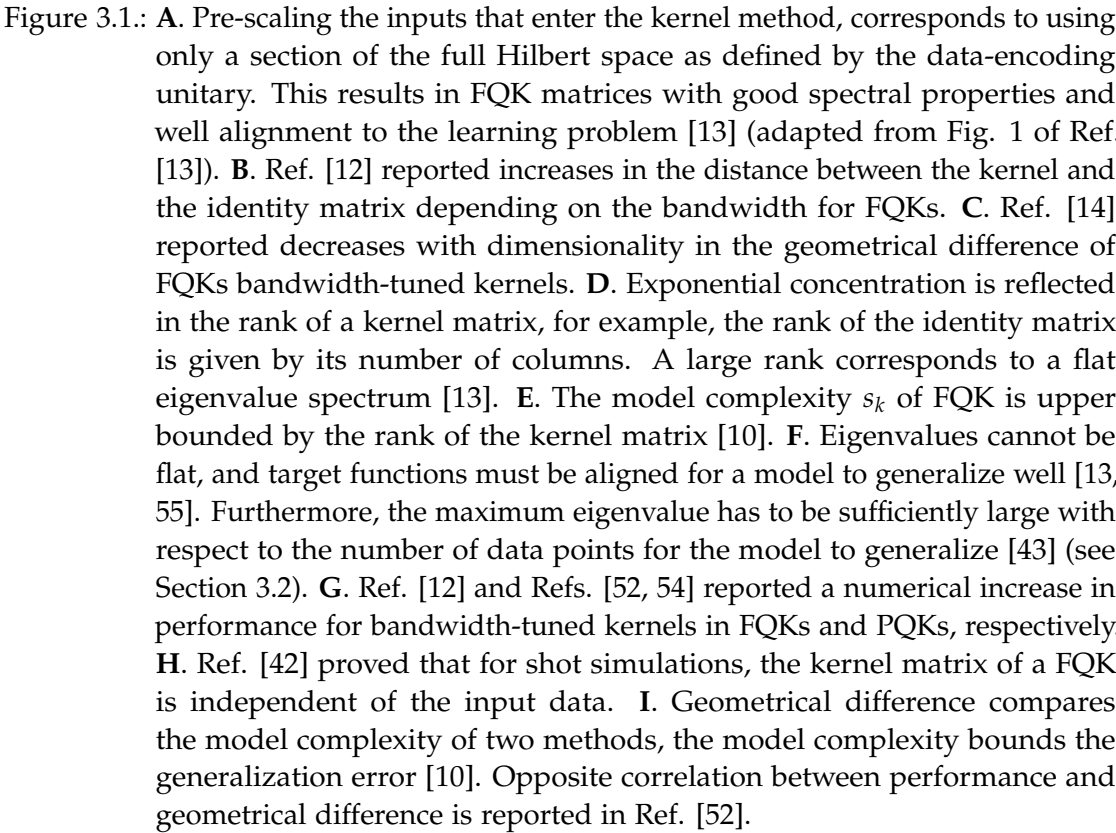
3. Methods

QKs methods generally compare points in an exponentially large Hilbert space, this can lead to the so-called problem of exponential concentration [10, 42, 43]. Similarities between points become exponentially small in sufficiently large inner product spaces. As a consequence, QKs do not generalize well, this is reflected in the **spectral and matrix properties** of the kernel [10, 42, 43]. Some alternatives to mitigate this problem have been suggested. For example, projecting the kernels to a semi-classical representation [10, 42, 43, 52] or using a quantum version of the Fisher Kernel [53], which takes into account the geometric structure of the data. Another approach involves rescaling the size of the points that enter the kernel model [12–14, 52, 54], a method known as **bandwidth tuning**. This technique has been shown to improve the generalization properties [12, 13] but at the same time results in QKs that are classically tractable [14]. The purpose of this chapter is to provide an explanation of each of these topics, which are condensed into Fig. 3.1.

3.1. Exponential Concentration

The problem of exponential concentration in kernels is similar to the well known problem of Barren Plateaus in QNNs [31, 32, 56–61]. Due to the size of the Hilbert space, loss gradients of the optimization landscape are exponentially small with system size. In the case of kernels, the inner products tend to zero but the loss landscape is convex and a solution is guaranteed to exist. However, obtaining inner products corresponds to statistical measurements, thus stochastic noise greatly affects the problem. As a final consequence, predictions made by a QK obtained with polynomial measurements in the regime of exponential concentration, are independent of the input training data [42]. In this section, we will quantitatively define the problem of exponential concentration and some of the design choices that lead to it.

Definition 6. Exponential Concentration [42]. *Consider a kernel $k(\mathbf{x}, \mathbf{x}')$, that can be measured on a quantum computer as the expectation value of some observable. Consider $\mu = \mathbb{E}_{\mathbf{x}, \mathbf{x}'}[k(\mathbf{x}, \mathbf{x}')] the mean value of the kernel entries.$*



- $k(\mathbf{x}, \mathbf{x}')$ is deterministically exponentially concentrated in the number of qubits n towards a certain x -independent value μ if

$$|k(\mathbf{x}, \mathbf{x}') - \mu| \leq \beta \in O(1/b^n) \quad (3.1)$$

for some $b \geq 1$ and $\forall \mathbf{x}, \mathbf{x}'$

- $k(\mathbf{x}, \mathbf{x}')$ is probabilistically exponentially concentrated in the number of qubits n towards a certain x -independent value μ if

$$\Pr_{\mathbf{x}, \mathbf{x}'} |k(\mathbf{x}, \mathbf{x}') - \mu| \geq \delta \leq \beta/\delta^2, \beta \in O(1/b^n) \quad (3.2)$$

for some $b \geq 1$ and $\forall \mathbf{x}, \mathbf{x}'$

By using Chebyshev's inequality¹ in Eq. 3.2, probabilistic exponential concentration can be detected by the variance of the kernel. If the kernel is probabilistic exponentially concentrated, then $\text{Var}[k(\mathbf{x}, \mathbf{x}')] \in O(1/b^n)$. The variance of a set of kernel entries can be calculated as,

$$\text{Var}[k(\mathbf{x}, \mathbf{y})]_{\mathcal{X}} = \frac{1}{N} \sum_{i,j} (k(\mathbf{x}_i, \mathbf{x}_j) - \mu)^2, \quad (3.3)$$

where N is the number of data points in the input set $\mathcal{X}$. Kernel values of identical points $k(\mathbf{x}, \mathbf{x})$ are always equal to 1, and not included in the variance calculation.

As a consequence of exponential concentration, QKs measured through the overlap of two states will resemble the identity matrix, with increasing number of qubits (Proposition 1 of Ref. [42]).

3.1.1. Sources of Exponential Concentration

Now, we briefly discuss some of the aspects that produce exponential concentration in QKs, as identified by Ref. [42].

- **Expressibility-induced concentration (FQK and PQK)** [42]: A data-encoding unitary $U(\mathbf{x})$, as defined in Eq. 2.28, for a given set of inputs $\mathbf{x} \in \mathcal{X}$ generates an ensemble of unitary matrices $\mathcal{U}_{\mathcal{X}} = \{U(\mathbf{x}) | \mathbf{x} \in \mathcal{X}\}$. The expressibility of U is defined by how close, its ensemble is from a 2-design² [42, 63]. When the ensemble resembles a uniform random distribution, $\mathcal{U}_{\mathcal{X}}$ is maximally expressive. Ref. [42] derived a bound for the variance of the kernel in terms of the expressibility of the data-encoding circuit. The more expressive the circuit, the tighter is the

¹Chebyshev inequality states that, for a random variable X , $\Pr|X - \mathbb{E}(X)| \geq \delta \leq \text{Var}[X]/\delta^2$

²See Ref. [62] for an introduction to quantum t -designs

bound. Intuitively, for very expressive data-encoding unitaries, an inner product between random vectors in an exponentially large space is calculated. These two random vectors are orthogonal with high probability, resulting in inner products very close to zero.

- **Entanglement-induced concentration (PQK)** [42]: As a way to mitigate exponential concentration, PQKs were introduced. By projecting to a semi-classical representation the dimension of the problem is reduced. However, the projection itself introduces exponential concentration. Intuitively, tracing out highly entangled states results in maximally mixed local states. Thanasilp *et al.* [42] show that the larger the entanglement entropy between a reduced state and a maximally mixed state, the more concentrated the kernel is. Specifically, the deviation of the kernel from a fixed value of 1 is bounded by the entanglement entropy.
- **Global-measurement induced concentration (FQK)** [42]: A global measurement is a measurement that acts on all qubits of a quantum state. They are a defining characteristic of FQKs. Global-measurements again are problematic when an exponentially large Hilbert space is available. Ref. [42] showed that even for product encoding feature maps with low expressibility and no entanglement, extracting information of objects that are global in this exponentially large space leads to exponential concentration. A bound for the variance of the kernel in terms of the expressibility of **local** data-dependent unitaries was derived.
- **Noise (FQK and PQK)** [42]: Current hardware is noisy, this worsens all previous sources of exponential concentration. It was shown that the variance of noisy kernels is bounded by a noise parameter dependent term and that the noisy state concentrates to the maximally mixed state [42].

Exponential concentration leads to kernels which require exponentially many shots to be meaningful. Moreover, even in the case of statevector simulations (without shots), this problem leads to ill-conditioned kernel matrices that may not generalize well [10, 43]. In the next section, we will discuss the generalization properties of QKs.

3.2. Generalization on kernel methods

How well does a (classical or quantum) kernel method make predictions on unseen data? The main theoretical tools to answer this question are based on studying the eigenspectrum of the respective kernel matrices. Fundamentally, kernels provide a basis expansion to describe the function that we want to learn.

3.2.1. Spectral Bias and Task-Model Alignment

By using the replica method [64] of statistical mechanics, Ref. [55] derived an expression to understand the generalization error, $E_g = \mathbb{E}_{x \sim \mathcal{D}} |(f(x) - \bar{f}(x))^2|$, i.e. the MSE of the out-of-sample predictions between the estimator function $f(x)$ and the true target $\bar{f}(x)$. The generalization error is given by,

$$E_g = \left\langle \int dx p(x) (f(x) - \bar{f}(x))^2 \right\rangle_{\mathcal{D}}, \quad (3.4)$$

with $p(x)$ the distribution of the dataset³. Their results depend on decomposing the kernel matrix into its orthogonal eigenfunctions $\Phi_l(x)$ and eigenvalues η_l using Mercer's Theorem [24] (Appendix A.3):

$$\eta_l \Phi_l(x') = \int k(x, x') \Phi_l(x) p(x) dx = \frac{1}{N} \sum_{i=1}^N k(x_i, x') \Phi_l(x_i). \quad (3.5)$$

In this context, the estimator and the target function can be written in terms of the basis set $\sqrt{\eta_l} \Phi_l(x)$, then,

$$f(x) = \sum_l a_l \Phi_l(x) = \sum_l w_l \sqrt{\eta_l} \Phi_l(x), \quad \bar{f}(x) = \sum_l \bar{a}_l \Phi_l(x) = \sum_l \bar{w}_l \sqrt{\eta_l} \Phi_l(x), \quad (3.6)$$

where the norm is given by $\|f\|^2 = \sum_l a_l^2 = \sum_l w_l^2 \eta_l$ and w_l and $\bar{w}_l$ are the weights of the estimator and the target function projection onto the basis $\sqrt{\eta_l} \Phi_l(x)$ ⁴. Using the $\sqrt{\eta_l} \Phi_l(x)$ decomposition, it was derived that the generalization error in the noiseless regime for a regression task, can be expressed as a sum of the contributions from each eigenmode,

$$E_g = \sum_l \eta_l \bar{w}_l^2 e_l \quad (3.7)$$

$$= \sum_l \eta_l \bar{w}_l^2 \underbrace{\left(\frac{1}{\bar{w}_l^2} \langle (w_l - \bar{w}_l)^2 \rangle_{\mathcal{D}} \right)}_{e_l} = \sum_l \eta_l \bar{w}_l^2 \left(\frac{1}{1 - \gamma} \frac{\kappa^2}{(\kappa + P \eta_l)^2} \right) \quad (3.8)$$

with constants $\kappa = \lambda + \sum_l \kappa \eta_l / (\kappa + P \eta_l)$, and $\gamma = \sum_l P \eta_l^2 / (\kappa + P \eta_l)^2$. From this equation, Ref. [55] proposed two ideas to understand generalization in kernel methods,

Claim 1. Spectral Bias [55]. *Kernel eigenfunctions Φ_l with large eigenvalues η_l can be estimated with kernel regression using a smaller number of samples.*

³In the case of a real dataset, it can be assumed $p(x) = \frac{1}{N} \sum_{i=1}^N \delta(x - x_i)$.

⁴and a_l and $\bar{a}_l$ are the projection coefficients onto the basis $\Phi_l(x)$

To gain some intuition, notice from Eq. 3.8, that larger eigenvalues η_l correspond to smaller modal errors E_l , i.e. $\eta_l \geq \eta_{l'} \implies E_l \leq E_{l'}$ [55]. Now, let us consider two cases of eigenvalues spectrum,

1. Decaying eigenvalues spectrum: consider that the eigenvalues are ordered in decreasing order, $\eta_1 > \eta_2 > \dots$. In this situation, the larger the eigenvalue η_l , the smaller the modal error E_l , therefore modes with larger eigenvalues are easier to learn. On the other hand, large modal errors do not pose a problem as they are normalized and their contribution to the generalization error can be suppressed by the small η_l .
2. Flat eigenvalue spectrum: consider that all eigenvalues are roughly equal, $\eta_1 \approx \eta_2 \approx \dots$. Here, all modes have approximately the same modal error E_l and all contribute equally to the generalization error because the eigenvalues are similar. Thus, learning all modes is equally hard.

Hence, a kernel with a decaying spectrum and large initial modal eigenvalues will exhibit a smaller generalization error compared to a flat spectrum, on the same dataset and as the number of training points increases. A consequence of this result is that for understanding the generalization properties of a kernel it is important to track the amount of information of the target function contained in the first l modes. To analyze the compatibility between the target function and the kernel, Ref. [55] introduced the concept of **Task-Model Alignment**.

Claim 2. Task-Model Alignment [55]. *A target function with most of their information in the eigenfunctions corresponding to the largest eigenvalues can be estimated efficiently and is compatible with the chosen kernel. The resulting task-model alignment can be measured by,*

$$C(l) = \frac{\sum_{l' \leq l} (\bar{w}_{l'} \sqrt{\eta_{l'}})^2}{\sum_{l'} (\bar{w}_{l'} \sqrt{\eta_{l'}})^2} \quad (3.9)$$

which quantifies how much of the projections coefficients $\bar{w}_{l'} \sqrt{\eta_{l'}}$ of the target function f are contained in the first l modes. The slower $C(l)$ saturates, the more the function is spread out in the eigenfunctions.

In Fig. 3.2, we illustrate the projection of the ideal target function f onto the eigenfunctions of two kernels. For a badly aligned kernel, the coefficients are spread out in the eigenfunctions, while for a well aligned kernel, the coefficients are concentrated in just the first modes. In Fig. 3.3, we apply these tools to a paradigmatic classification model. The kernel with less "flat" spectrum and steeper behavior of C_l achieves a higher accuracy as a function of number of training points. In Ref. [55], the authors numerically corroborate the theory outlined above using literature datasets.

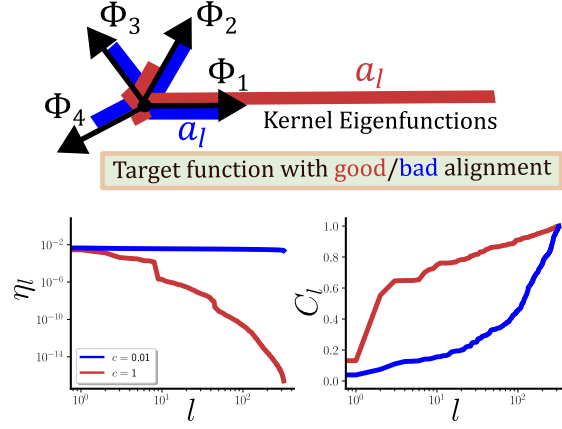


Figure 3.2.: Upper panel: Schematic diagram of the projections $a_l = \bar{w}_l / \sqrt{\eta_l}$ of two different target functions (blue/red) on the eigenfunctions spanned by a kernel. Here, only four eigenfunctions are drawn. Lower panel: Reproduction of the results of Fig. 1 of Ref. [55], task-model alignment and eigenvalues of two (quantum) kernels with good and bad generalization properties.

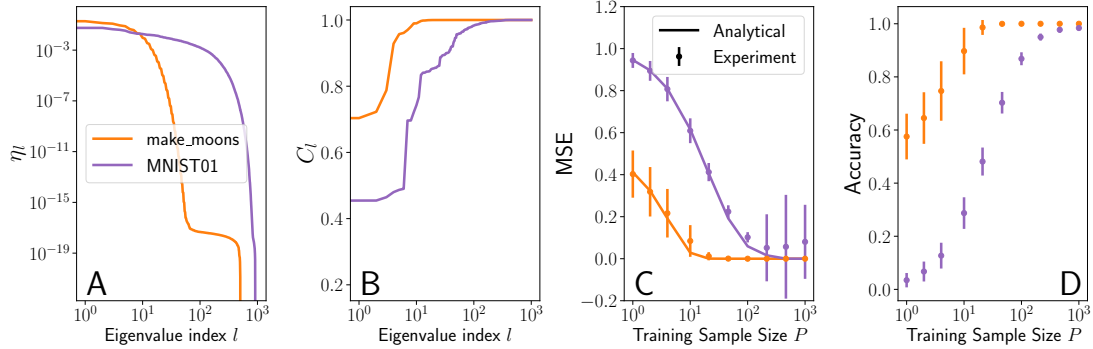


Figure 3.3.: Spectral Bias and Task-Model Alignment of an RBF kernel ($\gamma = 0.5$). Classification of the make_moons dataset [65] and MNIST [66] digits 0 and 1. From left to right: (A) eigenvalues η_l and (B) task-model alignment C_l as a function of the index l , (C) theoretical (solid lines) and experimental generalization (scattered points) error from Eq. 3.8 and (D) accuracy as a function of the number of training points P . Because the task-model alignment of the MNIST01 simulations rises slower than in make_moons, more samples are needed to learn the target function, resulting in smaller accuracy. Inspired by Fig. 1 of Ref. [55]

3.2.2. Maximum eigenvalue decay

Similar to the spectral bias, studying the behavior of the maximum eigenvalue of a (classical or quantum) kernel matrix provides information about the generalization of the model. In Ref. [43], it was shown that if the largest eigenvalue is small compared to the number of data points, the model will not be able to generalize (Theorem 1).

Theorem 1. Maximum Eigenvalue Decay from Ref. [13, 43] Theorem 3. *Let k be a valid kernel function. Let $\eta_{\max}$ be the largest eigenvalue of the corresponding integral operator of k . Suppose we have N training points $\mathbf{x}_i$ that are sampled from a probability distribution $p(\mathbf{x})$ and $y_i = \bar{f}(\mathbf{x}_i)$ for some square integrable target function $\bar{f}$. Then, for any $\varepsilon > 0$ with probability at least $1 - \varepsilon - \eta_{\max} N^4$, the generalization error is lower-bounded with,*

$$E_g \geq \left(1 - \sqrt{\frac{2\eta_{\max} N^2}{\varepsilon}}\right) \|\bar{f}\|^2. \quad (3.10)$$

Moreover, in the case of a FQK (Eq. 2.29), Ref. [43], showed that the maximum eigenvalue is upper bounded.

Theorem 2. Maximum Eigenvalue of a FQK is upper bounded from Ref. [43] Lemma 1. *The largest eigenvalue η_l of a FQK satisfies,*

$$\eta_{\max} \leq \sqrt{\text{Tr}(\rho_{\text{mean}}^2)}, \quad \rho_{\text{mean}} = \int \rho(\mathbf{x}) p(\mathbf{x}) d\mathbf{x}. \quad (3.11)$$

Note that for a FQK k , $\text{Tr}(\rho_{\text{mean}}^2) = \int k(\mathbf{x}, \mathbf{x}') p(\mathbf{x}) p(\mathbf{x}') d\mathbf{x} d\mathbf{x}'$.

$\rho(\mathbf{x})$ is given by the definition of FQK in Eq. 2.28. Also, $\text{Tr}(\rho_{\text{mean}}^2)$ is known as purity and measures the expressibility of the data-encoding [2, 43].

Using the bound of Theorem 2 in Theorem 1, Kübler *et al.* [43] show that for more than a certain number of qubits/dimensions n_0 , no function can be learned.

Theorem 3. FQK with n_0 qubits cannot learn any function from Ref. [43] Theorem 1. *Suppose $\text{Tr}\{\rho_{\text{mean}}^2\} \leq \delta < 1$, for a constant δ , as the number of qubits n grows. Suppose the sample size N grows polynomially in n , such that $N \leq n^l$ for a fixed l . Then, there exists $n_0 = n_0(\delta, l, \varepsilon)$ such that for all $n \geq n_0$, for a FQK, the generalization error is lower bounded with probability at least $1 - \varepsilon$,*

$$E_g \geq (1 - \varepsilon) \|\bar{f}\|^2. \quad (3.12)$$

Overall, the generalization properties can be understood by the spectral properties of the kernel. Similarly, Huang *et al.* [10], studied what kind of generalization advantages could kernel-based methods have on learning quantum models. In their research they connected generalization topics with the complexity of a learning problem.

3.2.3. Geometrical Difference

Ref. [10] derived generalization bounds that depend on the kernel entries. Based on the bounds, they defined a metric to compare the model complexity of two kernel models.

Specifically, they derived an analytical expression, for the error of a (classical or quantum) kernel model $f(\mathbf{x})$ to learn a target function $\tilde{f}(\mathbf{x})$ produced by a quantum neural network. The target function $\tilde{f}(\mathbf{x})$ is given by,

$$\tilde{f}(\mathbf{x}) = (\langle 0|^{\otimes n} U_{\text{enc}}^\dagger(\mathbf{x})) |U_{\text{QNN}}^\dagger(\boldsymbol{\theta}) \hat{O} U_{\text{QNN}}(\boldsymbol{\theta})| (U_{\text{enc}}(\mathbf{x})|0\rangle^{\otimes n}) \quad (3.13)$$

$$= \langle \mathbf{x} | U_{\text{QNN}}^\dagger(\boldsymbol{\theta}) \hat{O} U_{\text{QNN}}(\boldsymbol{\theta}) | \mathbf{x} \rangle \quad (3.14)$$

$$= \text{Tr}(U_{\text{QNN}}^\dagger \hat{O} U_{\text{QNN}} \rho(\mathbf{x})), \quad (3.15)$$

where $\hat{O}$ is a quantum observable and U_{QNN} is the unitary of the quantum neural network. Then, the subsequent theorem was proven.

Theorem 4. Prediction error of a kernel method learning a quantum model[10].

Let $\mathcal{D}$ be a dataset of N pairs $\{\mathbf{x}_i, y_i = \tilde{f}(\mathbf{x}_i)\}^N$. Consider the kernel model $f(\mathbf{x}) = \mathbf{w}^\dagger \boldsymbol{\phi}(\mathbf{x})$ (see Eq. 2.2) where $\mathbf{w}$ are the weights and $\boldsymbol{\phi}(\mathbf{x})$ is the feature vector for a given kernel $k(\mathbf{x}_i, \mathbf{x}_j) = K_{ij} = \boldsymbol{\phi}(\mathbf{x}_i)^\dagger \boldsymbol{\phi}(\mathbf{x}_j)$. The generalization error is given by,

$$\Delta E = \mathbb{E}_{\mathbf{x} \sim \mathcal{D}} |f(\mathbf{x}) - \tilde{f}(\mathbf{x})| \leq b \sqrt{\frac{s_K(N)}{N}} \quad (3.16)$$

with b a positive constant. The quantity s_K is referred to as model complexity and is defined as,

$$s_K(N) = \|\mathbf{w}\|^2 = \sum_{i=1}^N \sum_{j=1}^N (K^{-1})_{ij} \text{Tr}(U_{\text{QNN}}^\dagger \hat{O} U_{\text{QNN}} \rho(\mathbf{x}_i)) \text{Tr}(U_{\text{QNN}}^\dagger \hat{O} U_{\text{QNN}} \rho(\mathbf{x}_j)). \quad (3.17)$$

A small s_K means that the kernel model generalizes better to unseen points from $\mathcal{D}$. As the number of training points N increases, the bound is tighter [10].

In the specific case that the kernel model is given by a FQK (Eq. 2.29), it can be shown that $s_K \leq \min(d, \text{Tr}(\hat{O}^2))$, where $d = \text{rank}(K) \leq N$. Thus, Eq. 3.16 simplifies to,

$$\Delta E = \mathbb{E}_{\mathbf{x} \sim \mathcal{D}} |f(\mathbf{x}) - \tilde{f}(\mathbf{x})| \leq \kappa \sqrt{\frac{\min(d, \text{Tr}(\hat{O}^2))}{N}} \quad (3.18)$$

The rank of the kernel matrix represents the effective dimension of the original problem in the space spanned by the map used for the FQK. The model complexity s_K allows us to compare two different (quantum or classical) kernel models. For example, consider the following asymmetric distance between two models K_1 and K_2 ,

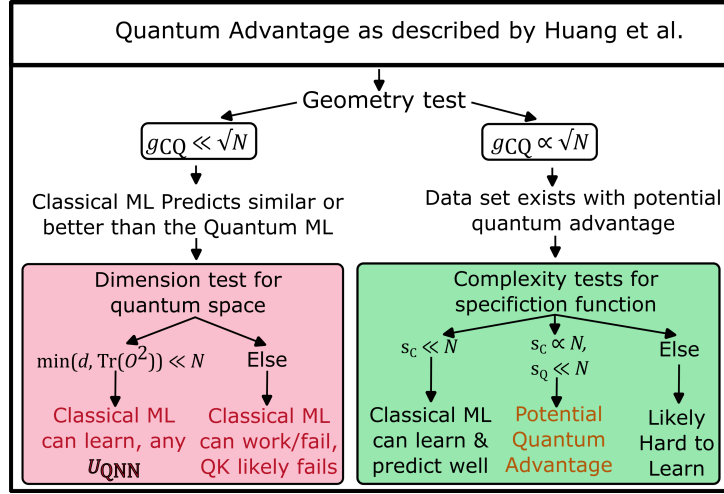


Figure 3.4.: Flowchart to identify potential quantum advantage in kernel methods. Adapted from Fig. 1B of Ref. [10]

$$g_{12} = g(K_1 \| K_2) = \sqrt{\left\| \sqrt{K_2} (K_1)^{-1} \sqrt{K_2} \right\|_{\infty}}, \quad (3.19)$$

where $\|\cdot\|_{\infty}$, corresponds to the spectral norm. If information about the regularization is included [10], the difference is given by:

$$g_{12} = g(K_1 \| K_2) = \sqrt{\left\| \sqrt{K_1} \sqrt{K_2} \sqrt{K_2} (K_2 + \lambda I)^{-2} \sqrt{K_1} \right\|_{\infty}}, \quad (3.20)$$

Using Eq. 3.19, it can be shown that,

$$s_{K_1} \leq g_{12}^2 s_{K_2}. \quad (3.21)$$

Thus, if g_{12} is small, the prediction performance of K_1 , should be similar or better to the model corresponding to K_2 . Let the first/second model be a classical/quantum model, respectively, i.e. $K_1 = K_c$ and $K_2 = K_q$, then, for a small g_{cq} , the classical model will have performance advantage. With these quantities, Huang *et al.* [10] devised a flowchart to identify potential quantum advantage, which we reproduce in Fig. 3.4.

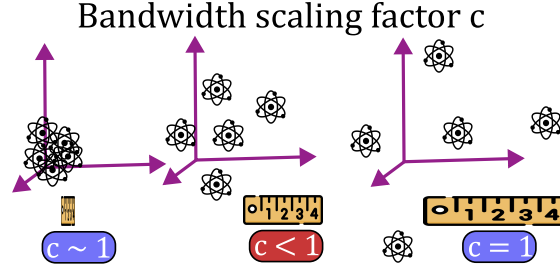


Figure 3.5.: Cartoon representing the effect of rescaling the size of the data points in the feature space of the kernel. Shaydulin & Wild [12] reported that $0 \leq c \leq 1$ improves model accuracy.

3.3. Bandwidth tuning

Choosing the correct hyperparameters of a ML model is a fundamental step for good performance. Shaydulin & Wild [12] numerically discovered that rescaling the input data that enter a FQK model has a beneficial effect. This means that changing x according to $x \rightarrow cx$, with a rescaling bandwidth factor c has proven to show a similar effect to setting the correct length scale (see Fig. 3.5) in classical kernels. They justified this by showing that QKs respond to bandwidth tuning in terms of the number of support vectors, similar to classical kernels. An increase in the prediction accuracy (see Fig. 3.6) and mitigation of the exponential decay of kernel values was reported (they found that there is an optimal $c = c^*$, which results in a kernel matrix very far away from the identity matrix, i.e. gives a maximal $\|K - \text{Id}\|^2$). Moreover, similar performance increases have been reported for linear [54] and RBF [52] bandwidth-tuned PQKs.

3.3.1. Generalization of bandwidth-tuned kernels

A theoretical explanation of the effect of bandwidth tuning was provided by Canatar *et al.* [13]. They showed that rescaling the size of points effectively limits the available states in the Bloch sphere. By limiting the expressibility of the quantum model, generalization is possible. To show this, they studied the data-encoding circuit of Eq. 2.36, and exactly calculated the eigenvalues of the kernel integral operator T_k (Eq. A.6). The eigenvalues go from a flat to a decaying spectrum with proper alignment as a function of the bandwidth. Moreover, they numerically extend this to other encoding maps and real datasets. The authors discuss that according to the theory of kernel generalization by Refs. [43, 55], the models should generalize (see Section 3.2.1).

In the specific case of the single-qubit kernel of Eq. 2.36 for a uniform distribution

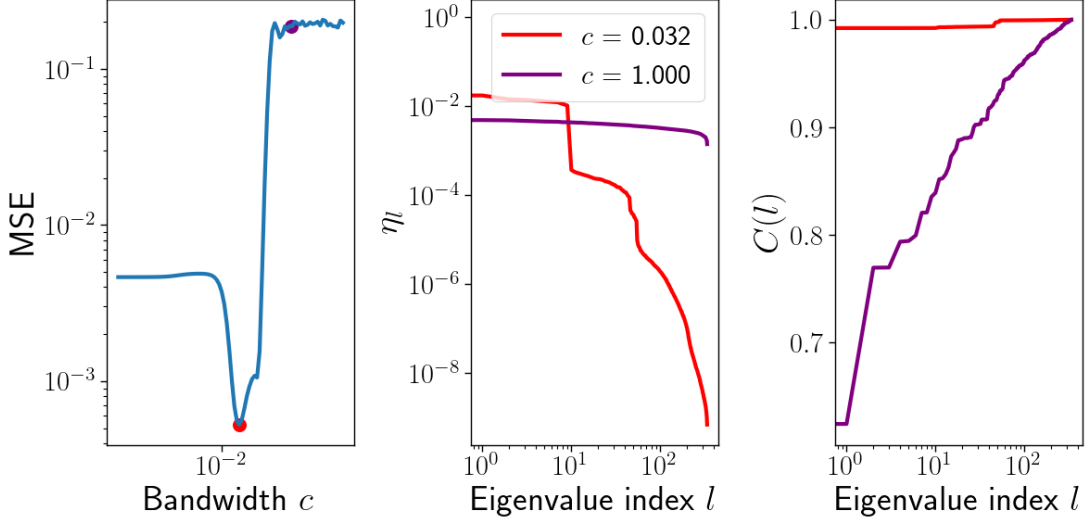


Figure 3.6.: Effect of bandwidth tuning in a toy dataset, there is an optimal value of c that minimizes the generalization error. From left to right: E_g as a function of bandwidth c , eigenvalue spectra η_l and task-model alignment C_l for two different bandwidths. Inspired by Fig. 1 of Ref. [12] and Fig. 2 of Ref. [13]. Numerical Setting: data-encoding unitary of Eq. 2.36 with $n = 9$ qubits and 3 re-uploading layers, 375 training and 225 testing points. Learning problem with a noisy target function $\hat{f}(x) = e^{-\|x\|^2/9^2} + 0.05\epsilon$, ϵ corresponds to uniform sampled noise between 0 and 1. Each training vector is sampled from a uniform distribution $p(x \in [-\pi, \pi]^{\times 9}) = 1/2\pi$.

$p(x \in [-\pi, \pi])$, they found the following eigenvalues for T_k ,

$$\lambda_1 = \frac{3}{8} + \frac{1}{8}\text{sinc}(2\pi c) + \frac{1}{8}\sqrt{(1 - \text{sinc}(2\pi c))^2 + 16\text{sinc}(\pi c)^2}, \quad (3.22)$$

$$\lambda_2 = \frac{1}{4} - \frac{1}{4}\text{sinc}(2\pi c) \quad (3.23)$$

$$\lambda_3 = \frac{3}{8} - \frac{1}{8}\text{sinc}(2\pi c) + \frac{1}{8}\sqrt{(1 - \text{sinc}(2\pi c))^2 + 16\text{sinc}(\pi c)^2}, \quad (3.24)$$

$$\lambda_4 = 0 \quad (3.25)$$

where $\lambda_1 > \lambda_2, \lambda_3, \lambda_4$ and $c \in [0, 1]$. For n qubits and independent distributions for each coordinate, the eigenvalues generalize to,

$$\eta_{n_1 n_2 n_3 n_4} = \lambda_1^{n_1} \lambda_2^{n_2} \lambda_3^{n_3} \lambda_4^{n_4}, \quad n_1 + n_2 + n_3 + n_4 = n, \quad (3.26)$$

If no bandwidth tuning is applied, $c = 1$, this results simplifies to,

$$\eta_{n_2 n_3} = 2^{-n-n_2-n_3} \quad (3.27)$$

Such that, for $n \gg 1$, $\eta_{n_2 n_3}$ is a flat spectrum. Thus, by Theorem 1, the kernel will not be able to generalize. Nevertheless, by adjusting c the eigenvalues can change. By finding the correct bandwidth scaling with the number of qubits, the spectra can decay. Moreover, they show that a properly bandwidth-tuned kernel would make Theorem 3 vacuous, i.e. only applicable with vanishing probability.

In brief, Ref. [13] showed that bandwidth tuning allows generalization by changing the spectral properties and expressibility of a QK. A natural follow-up research direction is to determine if bandwidth-tuned kernels can provide an advantage over classical methods in terms of accuracy.

3.3.2. Numerical Evidence against bandwidth-tuned kernels

The potential of bandwidth-tuned FQKs to provide a quantum advantage was recently investigated by Ref. [14]. The study found that for a sufficiently large number of qubits, the predictions of such models could be well-approximated by classically simulatable quantum models. The investigation involved calculating the geometrical difference [10], as defined by Eq. 3.19, with classical kernels and demonstrating that this geometrical difference decreases with increasing qubit count. In this section, we briefly describe the setup of their research.

Classification experiments were performed, where high dimensional problems were reduced to n -dimensions by doing Principal Component Analysis (PCA) [21, 22] on the features. Two data-encoding encoding unitaries were examined. Each data-encoding was compared against models based on the same circuits but without the quantum-like contributions.

The first data-encoding circuit, is inspired by the so-called Instantaneous Quantum Polynomial (IQP) of Ref. [11] and it was also used in the previous mentioned Refs. [12, 13]. The IQP circuit, implements the following mapping,

$$|\phi(\mathbf{x})\rangle = \hat{U}_Z(\mathbf{x}) \hat{H}^{\otimes d} \hat{U}_Z(\mathbf{x}) \hat{H}^{\otimes d} |0\rangle \quad (3.28)$$

with

$$\hat{U}_Z(\mathbf{x}) = \exp \left\{ \sum_{j=1}^d x_j \hat{Z}_j + \sum_{i,k=1}^d x_i x_k \hat{Z}_i \hat{Z}_k \right\}. \quad (3.29)$$

Table 3.1.: Summary of the main requirements for kernel generalization and their specific implications for FQKs.

Requirements for Kernel Generalization	Justification	In FQK
Maximum eigenvalue should not be small / Eigenvalues cannot be flat	Theorem 1 / Claim 1 due to Eq. 3.7	Theorem 3
Target function must be aligned with kernel	Claim 2	
Small model complexity $s_k = \ w\ ^2$	Theorem 4	Eq. 3.18

The "classical" model used to compare this kernel is given by,

$$|\phi(x)\rangle = \hat{U}_z(x) \hat{H}^{\otimes d} |0\rangle \quad (3.30)$$

as according to Ref. [14], although the output distribution of this data-encoding circuit is believed to be hard to reproduce, the FQK of the data-encoding circuit is classically simulatable in polynomial time [11, 67]. The second data-encoding unitary chosen is a 1D Heisenberg interaction map, which implements,

$$|\phi(x)\rangle = \left(\prod_{j=1}^d \exp(-i\hat{H}(x_j)) \right)^{n_l} |0\rangle, \quad (3.31)$$

with,

$$\hat{H}(x_j) = \hat{X}_j \hat{X}_{j+1} + \hat{Y}_j \hat{Y}_{j+1} + \hat{Z}_j \hat{Z}_{j+1} + x_j \hat{Z}_j \quad (3.32)$$

The classical counterpart of this map, removes the qubit interactions terms,

$$|\psi(x)\rangle = \prod_{j=1}^d \exp(-ix_j \hat{Z}_j) |0\rangle. \quad (3.33)$$

Within this framework, it was numerically shown, up to 22 qubits, that an increase in performance due to bandwidth tuning, corresponded to decreasing geometrical difference as the number of qubits increased. Similarly, Ref. [52] also found a correlation between small geometrical difference and top performance of bandwidth-tuned PQKs.

3.4. The complete picture

In Fig. 3.1, we summarize the main tools and their connections discussed in this chapter. Encoding classical information in an exponentially large space results in vanishing kernel entries [10, 42, 43]. This is the problem of **exponential concentration** [42],

which, as given in Definition 6 can be measured by the variance of the kernel. Another alternative is to measure how far apart the kernel matrix is from a diagonal matrix [12]. An exponentially concentrated kernel requires exponential number of measurements shots to be correctly trained and generalize [42]. Furthermore, even in the context of statevector simulations this phenomenon precludes **generalization** by affecting the matrix properties of a kernel [10, 43]. The main conditions for generalization are summarized in Table 3.1. These requirements are connected. Ref. [13] points out that for positive-definite kernels (symmetric matrices with non-negative eigenvalues, see Definition 8) with fixed trace, when the largest eigenvalue is small, the spectra will be flat. This will result in a large $\text{rank}(K)$ (see Theorem 3), as for such matrices, the rank is equivalent to the number of non-zero eigenvalues. This condition precludes generalization. To mitigate the problem of exponential concentration, **bandwidth tuning** was numerically discovered by Ref. [12] for FQKs. Refs. [52, 54] numerically demonstrated that this technique also works for PQKs. By using this method, an improvement in performance was seen and kernel matrices were far-away from the identity matrix. An explanation of this phenomenon was provided by Ref. [13], rescaling the points reduces the expressibility of the quantum model allowing for generalization by changing the spectral properties of the kernel. In a follow-up study, Ref. [14] found that for a sufficiently large number of qubits, bandwidth-tuned models lose the capacity for quantum advantage, as measured by the geometrical difference of Ref. [10].

4. Results

4.1. Bandwidth-tuned Kernels are numerically similar to polynomial kernels

It was shown that QKs suffer from exponential concentration, which can make learning inefficient [10, 42, 43]. Rescaling the size of the data points has been demonstrated to be an effective technique for increasing the performance of these methods [12] and allowing for good generalization [13]. However, it has also been numerically shown that such bandwidth-tuned QKs yield a small geometric difference [14, 52] when compared to QKs build from classically simulatable quantum circuits, implying that any quantum advantage is lost. The main contribution of this thesis is to explain why the bandwidth-tuned QKs become classically tractable. We show that the optimal rescaling factors are so small that the QKs yield similar properties to fully classical kernels, e.g. polynomial kernels, and that the entries of the resulting kernel matrices are very close. (including their variance) and lead to corresponding eigenvalue spectra that become qualitatively similar. We uncover these findings in detail by:

1. Reproducing previous results that show evidence of performance improvements [12–14] and a reduction in the geometrical difference [14, 52] through bandwidth tuning, with increasing qubit count. And, numerically demonstrating that, as the number of qubits increases, the difference between the QKs and polynomial kernels decreases. As can be seen by: (1) comparing their kernel matrix element by element, (2) their eigenvalue spectra, (3) the performance of the models, and (4) their geometrical difference.
2. Showing that for small bandwidths, in the examples studied, the numerical exponential concentration of the kernel always follows a polynomial decay (as a function of bandwidth), independent of kernel choice or dataset. We explain this phenomenon by exactly calculating the variance of the kernel for a toy model and capturing the same experimental decay by taking a fourth-order Taylor expansion. Optimal performance frequently corresponds to the regime where exponential concentration can be explained by a polynomial approximation.

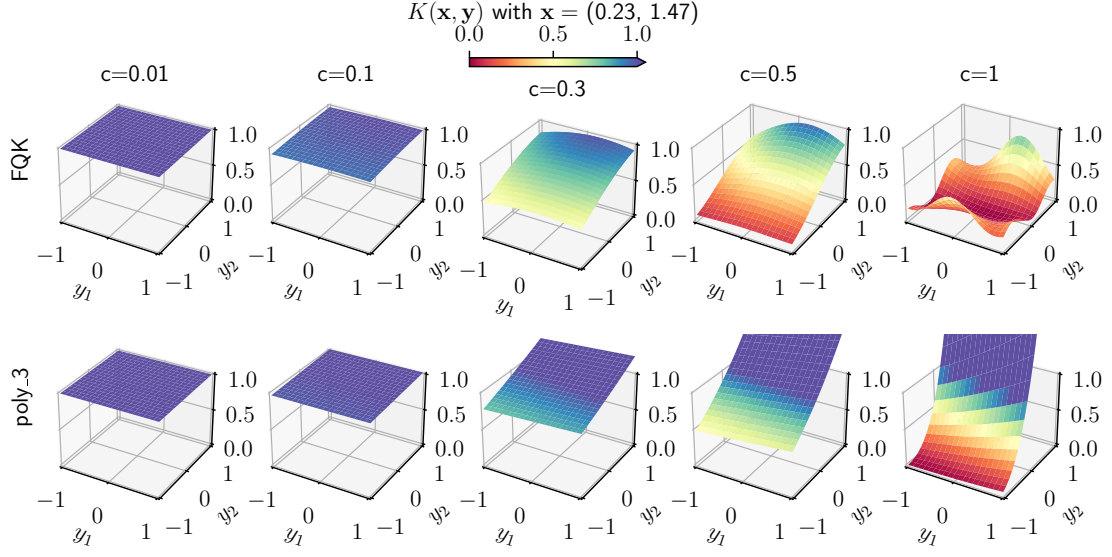


Figure 4.1.: Three-dimensional plot of the kernel matrix based on the IQP encoding circuit, Eq. 3.28 (upper panel) and a classical polynomial model, Eq. 4.1, of degree $d = 3$ (lower panel) for different rescaling values c . The first kernel entry x is fixed at $(0.23, 1.47)$ and the other entries are uniformly sampled for $y \in \mathbb{R}^2$. The color scale and the height of the surface correspond to the kernel value.

4.1.1. Motivating Example

As an introduction to the topic, we present a simple example that illustrates how quantum and classical models become similar with decreasing bandwidth parameter c .

QKs can be viewed as multi-dimensional Fourier series, as described by Eq. 2.37. In this picture, every trigonometric function of the series can be expressed exactly by a Taylor expansion. If each term is rescaled by a small rescaling factor c , higher-order contributions are minor and can be neglected. In this scenario, the effective kernels would be sufficiently approximated by lower-order terms of the Taylor expansion, which motivated us to compare the quantum models to polynomial kernels. The resulting matrices will not be exactly equal, as the polynomial models do not have the same functional form as the QKs (contrast Eq. 4.1 with Eq. 2.37). Nevertheless, by comparing with polynomial kernels of different orders, we gain insights about the actual leading order contributions of QKs in the small bandwidth regime. A visual intuition can be seen in Fig. 4.1, where we report two-dimensional kernel matrices for a quantum and a classical model for different bandwidths. As the rescaling factor

decreases, the contributions of the higher-order terms, in both methods, become less significant. Particularly, for $c = 0.1$, both kernel matrices represent almost planes in this low dimensional example. Starting from this point, we performed a series of experiments to reveal the similarity between quantum and classical models for optimal bandwidths in high dimensions.

Experimental Setup

We choose a similar setting to Refs. [12, 14] and report investigations on two datasets (see Section A.4 for technical details): `plasticc` [68] and `kmnist28` [69]. The original problem is transformed to an n -dimension dataset using PCA. We sample 400 points from the original datasets and split 80%/20% for training/testing. We perform classification experiments using SVC with quantum and classical methods for different rescaling factors c with a fixed regularization parameter $C = 514$. Each experiment is repeated 6 times with different seeds for sampling and splitting. Two data-encoding circuits are used, the IQP of Eq. 3.28 (Fig. A.1) and the 1D Heisenberg interaction (HamEvol) of Eq. 3.31 (Fig. A.2). The IQP circuit uses n qubits for a n -dimensional dataset, while the HamEvol circuit uses $n + 1$ qubits. In the following, we present results corresponding to FQKs and PQKs based on the IQP circuit for the `plasticc` dataset, as well as FQKs based on the HamEvol circuit for the `kmnist28` dataset. The remaining results of the experiments conducted in this thesis, which yield similar insights, can be found in the Appendix A.5.

4.1.2. Optimal polynomial kernels are very similar to optimal QKs for large number of dimensions

First, we reproduce¹ the results of Ref. [14] that show decreasing geometrical difference with increasing qubit count. Instead of using a classical version of the data-encoding circuits, as previously done in the literature [14], we compare the QKs to polynomial kernels of degree d (`poly_d`) as motivated above,

$$K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j) = (1 + \mathbf{x}_i \mathbf{x}_j)^d. \quad (4.1)$$

Here, we select a kernel of degree $d = 2$ and $d = 4$ for the comparison. This choice was motivated because numerical experiments showed good agreement. In Section 4.1.5, we complete our analysis including information of other degrees and RBF kernels. For

¹There is a small difference in the geometric difference calculations. Ref. [14] used Eq. 3.19 and we include the regularization parameter $\lambda = 1/2C$ using Eq. 3.20

this initial discussion, we compare the optimal classical and quantum models by five measures:

1. Receiver Operating Characteristic Area Under the Curve (roc auc) score: a classification metric that takes into account imbalances in the dataset, where a score of 1 corresponds to optimal accuracy and 0.5 to random guessing [65].
2. Largest eigenvalue $\eta_{\max}$: characterizes the generalization properties by Theorem 1 and Eq. 3.7.
3. Optimal Bandwidth c^* : rescaling factor used to multiply the input points. We report the value that produces the largest roc auc score.
4. Geometrical difference $g(K_C, K_Q)$: difference measurement between two matrices as defined by Eq. 3.20 that quantifies the possible advantages in generalization error. If $g \leq \sqrt{N}$, where N is the number of training points, no quantum advantage is possible.
5. Frobenius distance $F(K_1, K_2)$: distance measurement based on the Frobenius norm that quantifies the similarity between the kernel matrices,

$$F(K_1, K_2) = \|K_1 - K_2\|_F / \|K_1\|_F = \frac{\sqrt{\sum_{ij} k_1(x_i, x_j) - k_2(x_i, x_j)}}{\sqrt{\sum_{ij} k_1(x_i, x_j)}}. \quad (4.2)$$

In Fig. 4.2, we show experiments for the IQP circuit with `plasticc` (FQK and PQK) and `HamEvol` circuit with `kmnist28` (FQK). The box plots show each measure as a function of dimensionality. As the number of dimensions increases both methods improve their performance² (Fig 4.2 - first column). The two models (classical and quantum), show very similar roc auc scores. Furthermore, we observe that the larger the dimension and performance, the smaller the respective optimal bandwidth (Fig. 4.2 - third column). The same decrease of the rescaling factor is reflected in the geometrical difference between the ideal classical and QKs (Fig. 4.2 - fourth column), i.e. optimal QKs in larger dimensions correspond to smaller geometrical differences. The connections between optimal bandwidth, geometric differences and accuracy, as the number of dimensions increases, aligns with observations reported in the literature [14, 52]. To understand what renders the QKs tractable by classical kernels (decreasing geometrical difference), we compute the Frobenius distance $F(K_C, K_Q)$ (Eq. 4.2) between the classical and quantum kernels. This measure allows us to quantify the similarity between the

²These results, along with similar findings from Refs. [12, 13, 55], suggest that increasing the number of dimensions in a PCA-based classification problem decreases its difficulty and enhances the model's performance. Nevertheless, there are counter examples and this is problem dependent. See Section 4.1.6

kernel entries. With increasing dimensionality, the Frobenius distance between the quantum and classical models also decreases (Fig. 4.2 - fifth column). Thus, both kernels are not only similar in their generalization properties as measured by the geometrical difference, but they also define the same "distances" between the data points.

4. Results

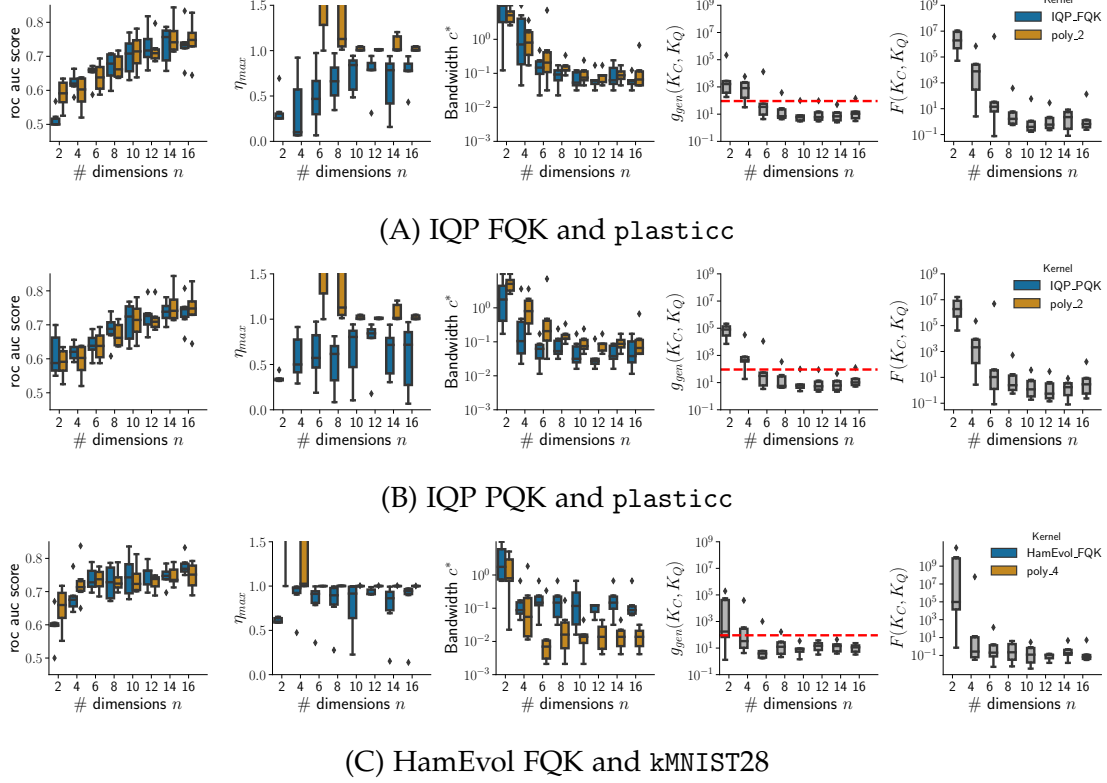


Figure 4.2.: Box plots of important quantities as a function of dimensionality, blue/yellow corresponds to the quantum/classical model and gray corresponds to the geometric difference $g(K_C, K_Q)$ and the Frobenius distance $F(K_C, K_Q)$ measures which depend on kernel matrices of both models. The boxes are constructed from the models with optimal bandwidths c^* , yielding the highest roc auc scores for each seed. Figures from left to right depict: roc auc score, largest eigenvalue $\eta_{\max}$, optimal bandwidth c^* , geometrical difference $g(K_C, K_Q)$ and Frobenius distance $F(K_C, K_Q)$. From top to bottom different models: (A) IQP FQK (plasticc), (B) IQP PQK (plasticc) and (C) HamEvol FQK (kMNIST28). The red dashed line for the geometrical difference corresponds to $\sqrt{N}$, the limit where no quantum advantage is possible. For increasing dimensions, the geometric difference and the Frobenius distance between the classical and quantum models decreases. This is reflected in similar performance and largest eigenvalue.

4. Results

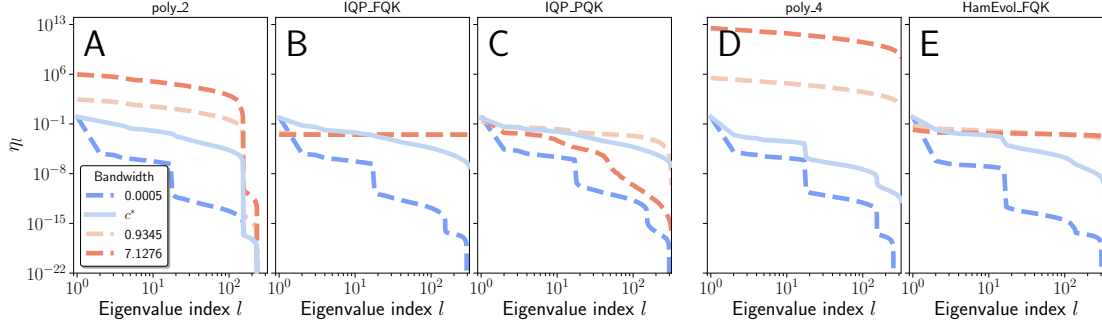


Figure 4.3.: Eigenvalues η_l of the kernel matrices. Different colors (cf. legend) correspond to different bandwidth tuning parameters and the number of dimensions is fixed to $n = 16$. The median optimal bandwidth is represented by " c^* " (light blue solid lines). The lines correspond to the median eigenvalues over the different seeds used. Each subfigure corresponds to a different model. From left to right: (A) `poly_2` ($c^* = 0.062$), (B) IQP FQK ($c^* = 0.062$) and (C) IQP PQK ($c^* = 0.032$) for the `plasticc` dataset, and, (D) `poly_2` ($c^* = 0.016$) and (E) HamEvol FQK ($c^* = 0.12$) for the `kMNIST28` dataset.

To further compare the generalization properties of both models, we can look at the largest eigenvalue $\eta_{\max}$ (Fig. 4.2 - second column ³). With increasing dimension, both maximum eigenvalues reach values $\sim O(1)$. The relation between $\eta_{\max}$, problem dimension and bandwidth, was intensively studied in Ref. [14]. If $\eta_{\max}$ decreases faster than $1/N^2$ the generalization error rate increases by Theorem 1 (Eq. 3.10). To maintain a fixed error rate, Slattery et al. [14] studied how changing the bandwidth leads to different maximum eigenvalues. The bandwidth was tuned, such that for increasing dimensions the maximum eigenvalue and error rate would remain constant. It was found that for a wide range of constant maximum eigenvalues, the optimal bandwidths would always lead to models with small $g(K_C, K_Q)$. In this work, we explicitly compare the eigenvalue spectra of the classical and quantum models for different bandwidths.

In Fig. 4.3, we report the eigenvalues as a function of their indices for the classical and quantum models. We fix the number of dimensions to $n = 16$ and report results for different bandwidths, with each color representing a distinct quantity. In Appendix A.5 - Fig. A.5 and Fig. A.6, we display the remaining spectra of the classical and quantum

³We limit the y -axis range of Fig. 4.2 - second column, to focus on how both models reach values $\eta_{\max} \sim O(1)$ for high dimensionality. Thus, some values for low dimensions in the case of the polynomials models are outside the plotting range.

4. Results

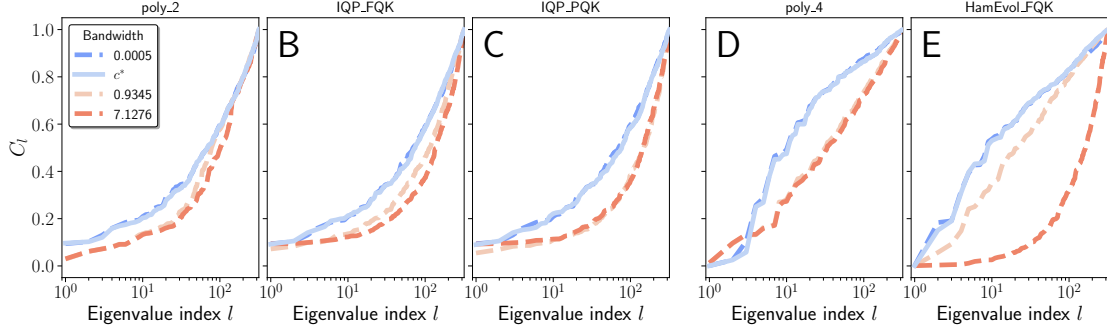


Figure 4.4.: Task-model alignment C_l of the kernel matrices. Different colors (cf. legend) correspond to different bandwidth tuning parameters and the number of dimensions is fixed to $n = 16$. Each subfigure corresponds to a different model. The median optimal bandwidth is represented by " c^* " (light blue solid lines). The lines correspond to the median C_l over the different seeds used. From left to right: (A) `poly_2` ($c^* = 0.062$), (B) `IQP_FQK` ($c^* = 0.062$) and (C) `IQP_PQK` ($c^* = 0.032$) for the `plasticc` dataset, and, (D) `poly_2` ($c^* = 0.016$) and (E) `HamEvol_FQK` ($c^* = 0.12$) for the `kMNIST28` dataset.

models for other dimensions.

For the large values ($c = 0.9345$ and $c = 7.1276$) plotted, the eigenvalues between classical and quantum models are distant by some orders of magnitude. In the FQK cases, these large bandwidths give rise to flat or almost flat spectra, as already reported in the literature [13, 43]. On the other hand, the PQK (Fig. 4.3C) shows a slight decay of the eigenvalues, even in this large bandwidth regime. This is expected as this was one of the original motivations behind the introduction of PQKs. As the bandwidth decreases in all QKs, the spectra show a steeper decay, which is consistent with the observation already done by Ref. [13]. Interestingly, for the optimal and smaller bandwidths, the eigenvalues of the quantum and classical models are very similar, both $\eta_{\max}$ are almost the same ($\sim O(1)$). Furthermore, for other indices l of these cases, the behavior is alike, especially in the other large eigenvalues that play a major role in the generalization properties of the model (see Section 3.2.1 - Claim 1).

The previous discussion extends to the task-model alignment C_l (Fig. 4.4)⁴, which quantifies the weight of the first l projected components of the eigenbasis (Eq. 3.9). We observe that bandwidth tuning positively affects task-model alignment in both

⁴One slight exception is the initial alignment of the optimal and smallest bandwidth of Fig. 4.4(D) with Fig. 4.4(E). We refer the reader to the second point in Section 4.1.6

quantum and classical models. Specifically, the task-model alignment at the optimal bandwidth shows a steeper rise compared to the other bandwidths. This implies, according to the spectral bias and task-alignment theory [55] (see Section 3.2.1), that less training data is needed to reach a certain performance level.

Fig. 4.3 and Fig. 4.4 suggest that looking at the kernel properties as a function of the bandwidth reveals new insights of where and how the kernels start to show polynomial-like behavior. This is the topic of the next section.

4.1.3. Models as a function of bandwidth

In the previous section, we compared the best performing quantum and classical models as a function of dimensionality and we observed that the optimal bandwidths of the quantum and the classical models decrease with dimensionality. Motivated by this, we now study how the properties of both models change as a function of c .

In Fig. 4.5, we show the average values over all the simulated seeds of the quantities of interest as a function of the bandwidth. We observe that decreasing the bandwidth, reduces the differences in both $g(K_C, K_Q)$ and $F(K_C, K_Q)$, to the classical models (Fig. 4.5 - third column and Fig. 4.5 - fourth column), as already hinted by the previous section. We also see that increasing the dimensionality of the problem shifts the differences g and F to the left in the respective plots. This means that for a fixed bandwidth, both $g(K_C, K_Q)$ and $F(K_C, K_Q)$ increase with the number of qubits.

Furthermore, as we have already seen, the similarities between models are reflected in other important properties of the kernels. These include the variance of the kernel entries, $\text{Var}[K]$, a key quantity to quantify the degree of exponential concentration in the kernel matrices (see Section 3.1), and the largest eigenvalue $\eta_{\max}$. In Fig. 4.5 - first column and Fig. 4.5 - second column, we report $\text{Var}[K]$ and $\eta_{\max}$, respectively, as function of the bandwidth for the classical and quantum models. By closely inspecting both quantities of the quantum models, we observe three distinct regimes for small-, intermediate-, and large-range c values.

First, we discuss the variance of the kernel entries. At **small** bandwidths, we observe a polynomial⁵ decay of the variance with respect to c for the quantum models, which is mostly unaffected by the number of dimensions. The decay corresponds to a line with a positive slope in the log scale. The polynomial kernels in this regime also show a similar slope but vertically shifted. In fact, the slope of the quantum model is proportional to c^4 , as it is discussed in Section 4.1.4. The small bandwidth regime is followed

⁵Note that Fig. 4.5 - first column and Fig. 4.5 - second column are given in log-scale

by the **intermediate** regime, where the slope of both classical and quantum models, starts to deviate. In this regime, the variance of the QKs peaks and starts decreasing, whereas the variance of the classical models continues increasing. Finally, for **large** bandwidths, the classical and quantum models are entirely different. The QKs reach constant values with regards to c , while the classical kernels diverge. Furthermore, both quantum models, FQKs and PQKs differ on the behavior of the variance. The FQKs show plateaus that exponentially decrease with the number of dimensions, as expected by the exponential concentration phenomenon. On the other hand, the PQKs show a second peak, that is followed by the plateaus. The plateaus once again decrease with dimensionality. The three regimes appear for all the QKs and distributions that we studied, hinting at a universal phenomenon.

Second, for the largest eigenvalue (Fig. 4.5 - second column), we observe the same three regimes. Similarly, for **small** bandwidths, we observe that $\eta_{\max}$ for the polynomial and QKs converges to 1. For **intermediate** bandwidths, $\eta_{\max}$ of the classical and quantum models starts to differ. For the QKs, $\eta_{\max}$ starts to decrease with increasing bandwidths, whereas for the polynomials it increases. Finally, for **large** bandwidths, we observe once again that the quantum and classical models are completely different and that the FQKs and the PQKs differ. The FQKs show plateaus that decrease with increasing dimensionality. This is connected to the exponential concentration phenomenon but expressed in the spectral properties, as related to Theorem 2 (see Ref. [43] Lemma 1). In contrast, the PQKs reach a minimum peak and then start to increase once more, before reaching a plateau. Note that the dependence between dimensionality and the $\eta_{\max}$ plateaus is different for the FQKs and the PQKs for large bandwidths. For FQKs, $\eta_{\max}$ decreases with increasing n , while for PQKs it increases.

By inspecting Fig. 4.5 - first column and Fig. 4.5 - second column, the connection between the exponential concentration phenomenon, as measured by the variance, and the largest eigenvalue of the kernel matrix is clear: the three distinct regimes appear in both quantities.

An insightful comparison about the optimal quantum models can be made by contrasting the boxes of the optimal bandwidths of Fig. 4.2 - third column, with the respective three regimes of the variance and the largest eigenvalue of Fig. 4.5. We observe that the optimal rescaling values mostly fall on the small bandwidth regime and on the transition from the small to the intermediate bandwidths. Suggesting once again, that the success of the bandwidth-tuned quantum models, for the datasets studied, can be attributed to the classical behavior of the kernels in these regimes.

To better understand the three variance regimes, in the next section we provide some toy model calculations that analytically reproduce the observed behavior of the FQKs.

4. Results

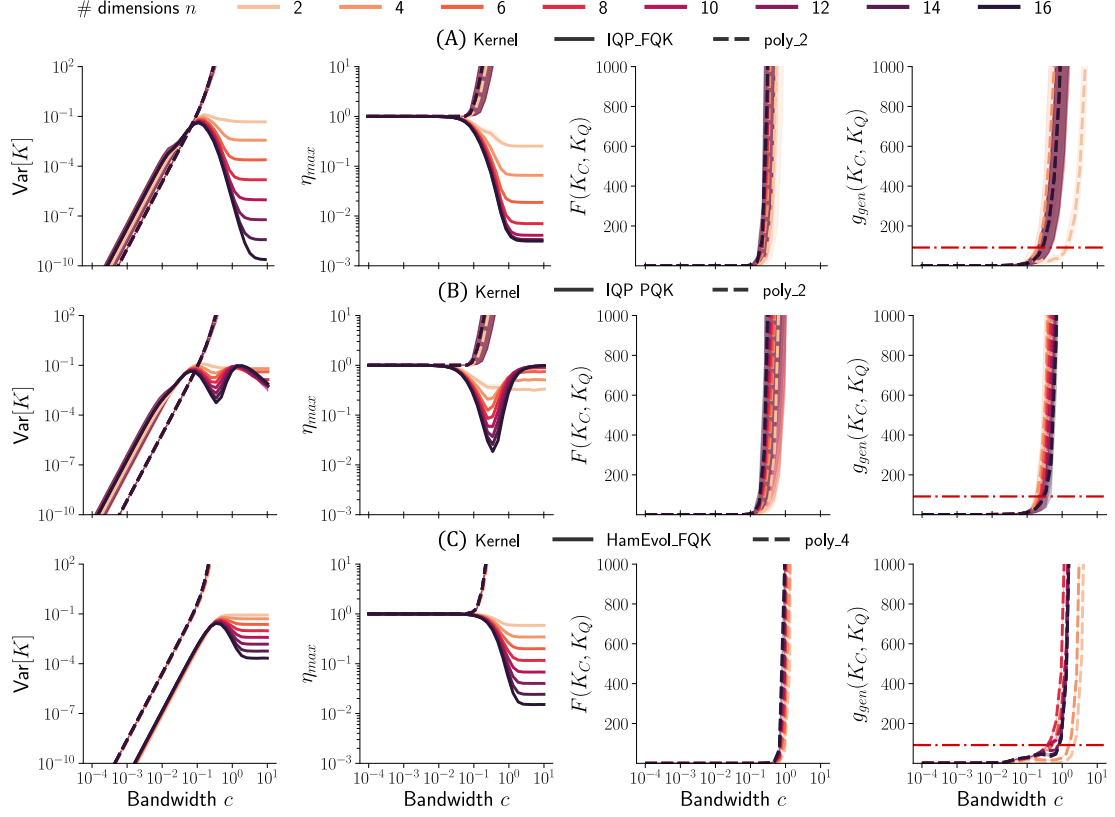


Figure 4.5.: Relevant measures as a function of bandwidth averaged over the different simulated seeds. Different colors correspond to different dimensionality, the darker the color the larger the number of dimensions. The shaded area corresponds to the standard deviation of the models. Dashed lines represent the classical models (either poly_2 or poly_4), whereas solid lines represent the quantum models. From top to bottom: (A) IQK (FQK) and plasticc, (B) IQK (PQK) and plasticc, (C) HamEvol (FQK) and kMNIST28. From left to right: variance of the kernel entries $\text{Var}[K]$, largest eigenvalue $\eta_{\max}$, Frobenius distance $F(K_C, K_Q)$ and geometrical difference $g(K_C, K_Q)$. The horizontal dot-dashed line in the geometrical difference plots corresponds to $\sqrt{N}$, the limit where no quantum advantage is possible. The variance and the largest eigenvalue of the quantum models exhibit three distinct regimes: small, intermediate and large bandwidths. In the small bandwidth regime, the classical and quantum models share the same behavior. The optimal bandwidths primarily occur within the small bandwidth regime and during the shift from small to intermediate bandwidths

4.1.4. Analytical investigations of limiting cases of the variance of the FQKs

As observed in our simulations (see Fig. 4.5 and Fig. 4.2) the optimal performance of QKs (for the datasets of this study) frequently occur for small bandwidths. To analytically track this regime, we propose to study a simple toy model that enables us to perform exact calculations.

We aim to model the dependence of the variance of the FQK entries on the rescaling parameter c and the dimensionality n . For this, we choose the Separable R_X data-encoding unitary, given by Eq. 2.36 (see Appendix A.6 for a complete derivation). This kernel has been very well studied in the literature with similar intentions [13, 36, 43].

Consider N training points x_i , where each x_i is sampled from the input set $\mathcal{X} \subset \mathbb{R}^n$, and every $x_i \in \mathcal{X}$ follows a probability distribution $p(x)$. We calculate the variance of the kernel as a function of the bandwidth. The variance of a random variable $k(x, y)$ in $\mathcal{X}$ is,

$$\text{Var}[k(x, y)]_{\mathcal{X}} = \mathbb{E}_{\mathcal{X}} [k(x, y)^2] - \mathbb{E}_{\mathcal{X}} [k(x, y)]^2 \quad (4.3)$$

where the expectation value is given by,

$$\mathbb{E}_{\mathcal{X}} [k(x, y)] = \int_{\mathcal{X}} k(x, y) p(x) p(y) dx dy \quad (4.4)$$

By exactly evaluating Eq. 4.3 for the simple Separable R_X kernel and assuming that each coordinate⁶ follows independent distributions, we can capture the three regimes of the variance of the kernel as a function of bandwidth. For these calculations, we assumed two different distributions (see Fig. A.9 in the Appendix), a uniform distribution and a raised cosine distribution.

For a uniform distribution $p(x) = 1^n = 1$ with $x \in \mathcal{X} = [0, 1]^{*n}$, we obtain

$$\text{Var}[k(x, y)]_{\mathcal{X}} = \left(\frac{3}{8} - \frac{\cos(Lc)}{(Lc)^2} - \frac{1}{16} \frac{\cos(2Lc)}{(Lc)^2} + \frac{17}{16(Lc)^2} \right)^n - \left(\frac{1}{2} - \frac{\cos(Lc)}{(Lc)^2} + \frac{1}{(Lc)^2} \right)^{2n} \quad (4.5)$$

Taking the small bandwidth limit $c \rightarrow 0$ in a fourth-order Taylor expansion, we find a polynomial approximation of the variance,

$$\text{Var}[k(x, y)]_{\mathcal{X}} \approx \alpha(n)(Lc)^4, \text{ with } \alpha(n) = 0.00243n. \quad (4.6)$$

⁶For Eq 2.36, we have $k(x, y) = \prod_i k(x_i, y_i)$ and assuming independent distributions for each coordinate, $p(x) = \prod_{i=1} p(x_i)$ implies $\text{Var}[k(x, y)] = \mathbb{E}_{\mathcal{X}} [k(x, y)^2] - \mathbb{E}_{\mathcal{X}} [k(x, y)]^2 = \prod_i \mathbb{E}_{\mathcal{X}} [k(x_i, y_i)^2] - \prod_i (\mathbb{E}_{\mathcal{X}} [k(x_i, y_i)])^2$

This leads to two important implications. First, in the small bandwidth limit, $\text{Var}[K]$ decays proportionally to c^4 and second, $\text{Var}[K]$ increases linearly in the number of dimensions n .

For the large bandwidth limit $c \rightarrow \infty$, we obtain

$$\text{Var}[k(x, y)]_{\mathcal{X}} \approx (a^n - b^n)/d^n, \text{ with } a = 3 \cdot 2^2, b = 2^3, d = 2^5, \quad (4.7)$$

i.e. $\text{Var}[K]$ becomes a constant that exponentially decays with the number of dimensions. Similar scaling behaviors in both c and n , for small and large bandwidths, are observed for the raised cosine distribution $p(x) = 1/2\pi (1 + \cos(x))$, as shown in Appendix 4.1.4.

These findings are displayed in Fig. 4.6 for both the uniform and cosine distribution for different number of dimensions by comparing numerical and analytical results. The numerics were obtained by sampling 1200 points from the respective distributions and calculating the variance of the kernel entries of FQK with Separable R_x encoding by the discrete sum of Eq. 3.3. Additionally, we show the limiting behavior calculated for the uniform (raised cosine) distributions with Eq. 4.6 (Eq. A.40) and Eq. 4.7 (Eq. A.41). The numerical experiments and analytical calculations align as expected for enough sample points, with the analytical limits accurately capturing the extreme behavior observed in the empirical results.

We emphasize that these calculations, based on very simple training distributions and a basic quantum circuit without entanglement or access to superposition, exhibit the same three regimes of variance in the kernel entries observed in the real experiments based on FQKs (Fig. 4.5A and Fig. 4.5C). These experiments, used non-classically simulatable QKs, i.e. Eq. 3.28 and Eq. 3.31, and non-trivial datasets (kMNIST28 and plasticc). In the small bandwidth regime, in both Fig. 4.5 and Fig. 4.6, we see the predicted scaling of c^4 . Furthermore, a very small linear scaling with the number of dimensions is seen in both figures. Meaning that in the small bandwidth regime, the variance of the kernel does not decrease with the number of dimensions, on the contrary, it slightly increases. This is in contrast to the large bandwidth regime, where the variance decreases exponentially with the number of dimensions. The current discussion applies mainly to FQKs. Although the real experiments with PQKs (Fig. 4.5B) exhibit the same behavior in the small bandwidth regime, the transition from intermediate to large values, as well as the behavior in the large regime, differs. Future work will extend the derivation to capture the observations of the PQKs.

In the next section we discuss again the connections between quantum models and classical models but taking a look at a broader range of classical kernels.

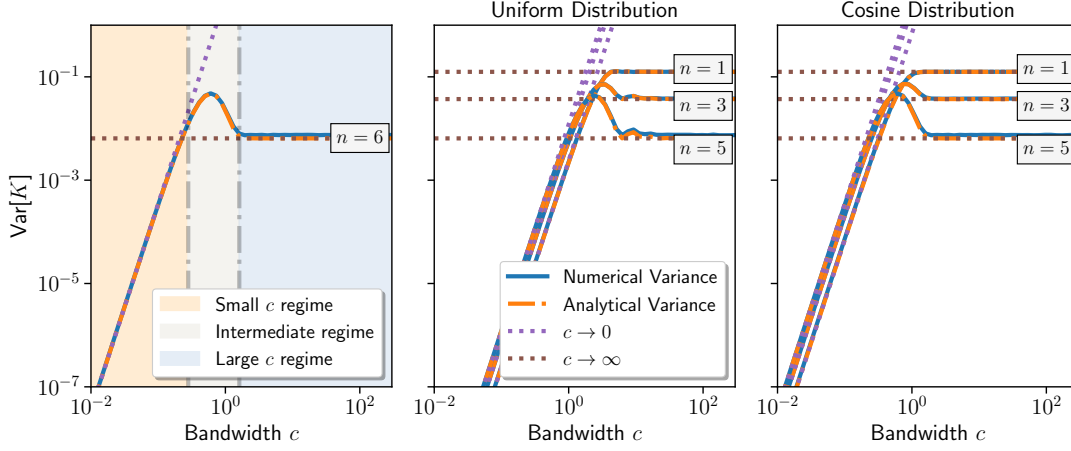


Figure 4.6.: Variance of the kernel as a function of bandwidth for the separable R_x encoding circuit based FQKs for two toy datasets. The first column, depicts the three regimes of the variance for the uniform distribution, with $n = 6$. The second/third column corresponds to the uniform/cosine distribution simulations. Three curves are shown for different number of dimensions $n = 1, 3, 5$. The numerics (blue solid lines) were obtained by sampling 1200 points from the corresponding distribution and calculating the variance of the kernel entries. For the analytical plots (orange dashed lines), we used the exact equations Eq. A.36 and Eq. 4.5, respectively. In dotted lines, we show the limiting behavior calculated with Eq. 4.6 and Eq. 4.7 for the uniform distribution, and Eq. A.40 and Eq. A.41 for the raised cosine distribution. As shown in Fig. 4.5 (first column), the experiments for real datasets using QKs based on encoding circuits with entangling gates show the same three regimes and limiting behaviors.

4.1.5. Comparing bandwidth-tuned quantum models with different classical models

The comparisons of sections 4.1.2 and 4.1.3 were made between polynomial kernels of fixed degree d and the QKs, without providing any explanation for the specific choice of classical kernel. Now, we compare at the same time, the distances between the quantum models and the polynomial kernels of degree $d \in \{1, 2, 3, 4\}$, as well as the RBF kernel. This will corroborate that QKs become similar to classical kernels for the optimal bandwidths for a wide range of classical models. Furthermore, by using different degrees of polynomial kernels, we can obtain a better understanding of the

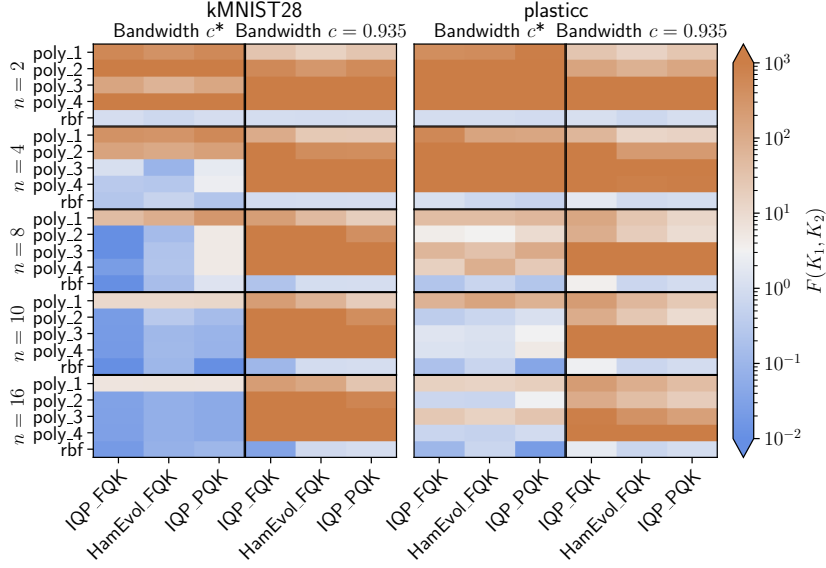


Figure 4.7.: Frobenius distances according to Eq. 4.2 heatmap between kernel matrices of the two datasets. Each block represents the distance between the quantum (on the x -axis) and classical (on the y -axis) kernel matrices. For each dataset two different bandwidths are shown. The c^* column corresponds to the matrices with optimal bandwidth (highest roc auc score), and the second column has $c = 0.935$. As the number of qubits increases, the distance in the c^* column decreases whereas in the other column continues to be large. The distances in kMNIST28 (left panel) are smaller than in plasticc (right panel). The RBF kernel is an exception as its values do not explode for c close to 1, see main text for a detailed discussion.

relevance of the higher order contributions of the QKs.

The differences between all classical and quantum kernels studied are displayed in Fig. 4.7 and Fig. 4.8. We report both $F(K_1, K_2)$ (Eq. 4.2) and $g(K_1, K_2)$ (Eq. 3.20) distances, for the optimal bandwidth c^* and c close to 1 (i.e. no bandwidth tuning). As the dimensionality increases, the distances in the c^* column are reduced but remain roughly the same for $c \sim 1$. These changes were similarly reported in the previous sections for $d = 2$ and the IQP circuit (Fig. 4.2A and Fig. 4.2B), or $d = 4$ and the HamEvol circuit (Fig. 4.2C).

In Fig. 4.7, for both the plasticc and kMNIST28 datasets with $n = 2$, the polynomial kernels are notably distant from the QKs. However, surprisingly, the RBF kernel is

not as far away. For $n = 4$, the simpler classical models remain distant from the QKs. Interestingly, though, the `poly_3` and `poly_4` begin to resemble the quantum models in the `kMNIST28` dataset. In contrast, for the `plasticc` dataset, the `poly_3` and `poly_4` are still quite distant from the quantum models. For `kMNIST28`, across higher dimensions, all non-linear polynomial kernels are very close to the quantum models, while for `plasticc`, this behavior is observed only when $n = 10$. Note that the difference between classical and quantum kernels depends on the choice of dataset. More intense red colors for the `kMNIST28` than in the `plasticc` dataset are seen. It is important to reemphasize that $F(K_1, K_2)$ will not be precisely equal to 0. The effective matrices are not the same because the polynomial models do not represent exactly the identical QKs functional form even in the small c limit (see second point in Section 4.1.6).

Similar general trends are observed in Fig. 4.8, but with different magnitudes. The only significant exception occurs for $n = 2$: we see a small geometric difference for c^* and `kMNIST28` with `poly_1` and `poly_3`, as well as for the FQK comparisons at $c = 0.935$ on both datasets. To understand this, we refer to Fig. 3.4, that suggest that a dimension test is necessary.

4.1.6. Limitations

In this chapter, we provided empirical evidence that bandwidth tuning gives rise to QKs that are similar to polynomial kernels. Therefore, explaining why such models cannot provide a quantum advantage. There are a couple of limitations to our study.

- Although we believe most of our results are general, our numerical investigation were done on a limited experimental study size. We only investigated two different classification datasets and two data-encoding circuits on similar framework to literature references [10, 12–14]. Moreover, both datasets had the unique property that increasing the number of dimensions in the PCA reduced the problem difficulty (as seen by increasing accuracy). This characteristic is not universal, see for example Fig. 6 of Ref. [38]. Nevertheless, even if the datasets do not show that characteristic, the difference between classical and quantum models decays exponentially with decreasing rescaling factor (Fig. 4.5). Thus, if the optimal bandwidth is small, the models will be in the regime of our experiments. Nonetheless, an open question still exists: does quantum advantage require datasets where optimal model performance is in a regime of moderate to large bandwidths? If true, what are the characteristics of these datasets?
- We claim that the bandwidth-tuned QKs are similar to the polynomial kernels. Nevertheless, one has to be careful with this statement. As even in the limit of

4. Results

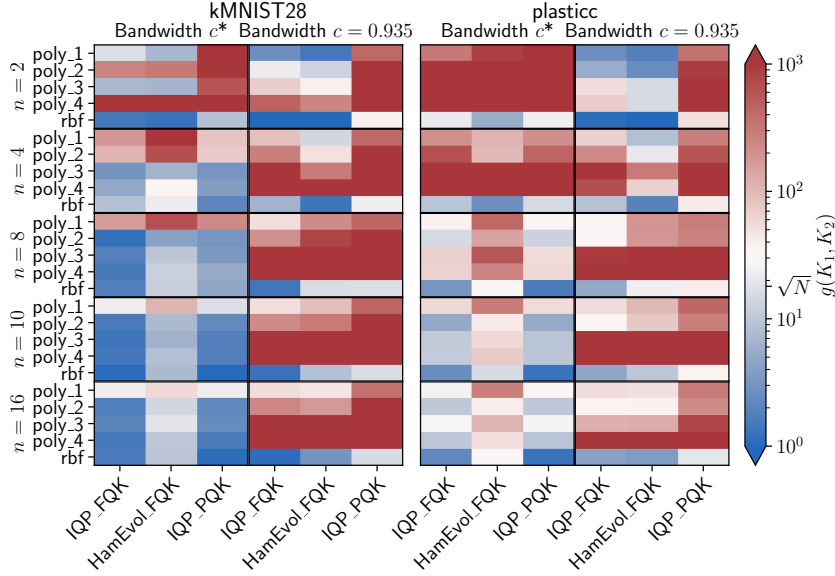


Figure 4.8.: Geometric difference according to Eq. 3.20 between kernel matrices of the two datasets. Each block represents the difference between the quantum (on the x -axis) and classical (on the y -axis) kernel matrices. For each dataset two different bandwidths are shown. The c^* column corresponds to the matrices with optimal bandwidth (highest roc auc score), and the second column has $c = 0.935$. As the number of qubits increases, the geometric difference in the c^* column decreases whereas in the other column continues to be large. The geometric differences in kMNIST28 (left panel) are smaller than in plasticc (right panel). The RBF kernel is an exception as it has a small geometric difference for almost all the blocks.

small c both models will not be the same. For instance, the small bandwidth limit of the FQK of Eq. 2.36 is

$$k(x, y) \stackrel{c \rightarrow 0}{=} 1 - \frac{c^2 (x - y)^2}{4} = 1 + \frac{c^2 xy}{2} - \frac{c^2 x^2}{4} - \frac{c^2 y^2}{4} \quad (4.8)$$

which, for example, is different from the polynomial kernel of $d = 2$:

$$k(x, y) = (1 + c^2 xy)^2 = c^4 x^2 y^2 + 2c^2 xy + 1 \stackrel{c \rightarrow 0}{\approx} 1 + 2c^2 xy. \quad (4.9)$$

At the same time, comparing the quantum models to the polynomial kernels allows us to quantify the importance of high order terms and how much the quantum elements really change models.

- Most of our discussion centered on polynomial kernels, because these kernels have clearly defined order of magnitude and empirically show the same behavior as the quantum inspired encoding maps used by Ref. [14]. In the sense that, as the number of dimensions increases, the difference decreases. However, RBF kernels always show a small geometric difference with regards to quantum models (see Fig. 4.8). Meaning that, although a given quantum model has a large g in regard to a polynomial kernel, it will not necessarily provide a quantum advantage as the RBF kernels, or possibly other classical models, will likely perform better. Note that, the similarity between the RBF kernels and QKs has been already been hinted in the literature, as reported by Ref. [38] (see Fig. 16 and Fig. 17 therein).
- One could argue that our results are a simple consequence of a small geometric difference between models, in the sense that, if $K_C(x_i, x_j)$ and $K_Q(x_i, x_j)$ are of the same order of magnitude, g_{CQ} should be small ~ 1 , according to Ref. [10]. However, we also numerically check the other direction. In large dimensions, small geometric differences correspond to small Frobenius distances, i.e. the classical and quantum models are capturing the same distances. Moreover, our investigation also looks into other relevant quantities ($\eta_{\max}$ and $\text{Var}[K]$) in the context of generalization and exponential concentration that are not trivially connected to the geometric difference.
- From a technical perspective, some points still need to be addressed. The FQK analytical model has to be quantitatively benchmarked with the real simulations to understand its own limitations. Additionally, the analytical study must be extended to capture the PQKs behavior, as well as more numerical experiments with PQK models need to be completed. Addressing these points may also help clarify the opposite relation between dimensionality and the largest eigenvalue of the kernel matrix for PQKs (Fig. 4.5B).

4.1.7. Discussion

Here, we have successfully reproduced and extended the phenomenon observed in the literature [14] that bandwidth-tuned QK become classically tractable. Furthermore, we have provided an explanation of why bandwidth tuning do not provide a quantum advantage by making a thorough comparison with the classical models (by computing g , F , eigenvalues and task-model alignment). We reveal insights into the mechanisms that lead QKs to transition to a regime where they are approximated by classical kernels. This was done by studying the QKs as a function of the bandwidth tuning parameter. In this respect, we derived analytical expressions for $\text{Var}[K]$, based on toy models,

that capture our numerical results and therefore foster our understanding of these properties.

Additionally, our investigations provide new insights into the bandwidth tuning technique discussed in previous research [12–14, 52]. For instance, Ref. [14] showed that, in their experimental setting, no quantum model could achieve a quantum advantage through bandwidth tuning. And we further address the open question of how quantum and classical bandwidth-tuned models compare.

Furthermore, our results provide a comprehensive perspective on bandwidth tuning within the context of QKs research (Fig. 3.1). Tuning the bandwidth alters the variance of the kernel entries across three distinct regimes. When no data is rescaled, the calculations occur in the large-bandwidth regime (e.g., $c = 1$). We demonstrate that the large-bandwidth regime exhibits an exponential decay in the number of qubits, as previously reported in the literature [10, 42, 43], whereas the small-bandwidth regime does not exhibit such decay. The absence of this decay was already hinted at in Ref. [12] (see Fig. 3A therein) but not fully characterized. This small bandwidth regime is important, as in our experiments it most frequently yields the best-performing models for large dimensions. We demonstrate that these regimes also manifest in the largest eigenvalue of the kernel matrix. When the maximum eigenvalue is not subject to exponential decay with system size, generalization in kernel methods is possible [13, 43, 55]. The absence of exponential decay in the maximum eigenvalue had been previously reported in Refs. [13, 14], here we contextualize it in the limiting values of the bandwidth, and show that the classical models have the same behavior for small bandwidths. Additionally, our experiments reveal that the intermediate and small-bandwidth regime correspond to regions where the geometric differences between models are small, consistent with the literature [14]. Therefore, we make connections of the exponential concentration phenomenon (as measured by the variance $\text{Var}[K]$) with the generalization properties of the model (g , eigenvalues and task-model alignment) in the context of bandwidth tuning. A connection between these topics had only been argumentatively [13] made through the behavior of the rank of the kernel matrix [10]. Furthermore, these results also provide an explanation of the findings of Ref. [52] (see Fig. 4 therein), where an opposite relation was found between good accuracy and large geometric difference for different bandwidths in the context of PQKs. This is, QKs improve their performance because they resemble classical models, producing small geometric differences.

Beyond that, our results align with recent trends in the QML community, which suggest that the absence of barren plateaus or exponential concentration implies classical simulability [44]. It has been argued and shown that some successful variational

quantum algorithms typically operate within classically simulatable subspaces [44]. For instance, quantum convolutional neural networks (QCNNs) have shown some success in classification tasks and were considered a promising method, particularly as they were found to be free of barren plateaus [58]. Nevertheless, it was recently shown that QCNNs are classically simulatable [70]. In our case, bandwidth tuning alleviates the consequences of exponential concentration and increases the performance of the model by making it more similar to classical models. Similarly, a large-scale benchmarking study of variational QML conducted by Ref. [38] revealed that classical methods consistently outperform variational quantum methods. Additionally, in accordance to our observations, the study showed that in some cases, QKs resemble classical kernels (see Fig. 16 in Ref. [38]).

A key question moving forward for variational QML is understanding what types of data are most suitable for learning in QCs, as highlighted by Refs. [38, 52]. Tuning the bandwidth can be seen as modifying the original dataset, and we observe that classical and quantum methods differ primarily based on the choice of bandwidth (dataset). Current research efforts largely focus on model design, aiming to develop techniques that are difficult to simulate classically, while often overlooking the dataset perspective. We believe that exploring the construction of artificial datasets or identifying challenging datasets is an important next step that is also closely connected to bandwidth tuning. A slightly underexplored idea, proposed by Ref. [10], involves artificially generating datasets by maximizing their geometric differences from classical models. Alternatively, artificial datasets that take advantages of the multifrequency nature of quantum models could be promising. For instance, Ref. [20] introduced a frequency degree of freedom (similar to a tuneable bandwidth) for QNNs and observed performance improvements, however the emphasis remained on model design rather than problem design. Perhaps, variationally optimizing the geometric difference by taking into account the multifrequency nature with bandwidth tuning could be a promising direction.

4.2. Differential Equations Solvers using Variational Quantum Machine Learning

As part of this master thesis, we also contributed to the development of a quantum machine learning library. sQUlearn [17] is a Python library that implements quantum machine learning algorithms with emphasis in variational QML and integration with classical machine learning libraries [65]. The package is mainly divided into two main components, the QNN engine and the QK engine. During this master thesis, two relevant contributions were made to the library. We implemented a differential equation solver based on (1) QKs [33] and on (2) QNNs [15]. A brief explanation of these implementations is given in Appendix A.7. Moreover, in order to better understand the capabilities of QKs solvers, we performed a small hyperparameter study.

4.3. A small hyperparameter study of QKs as differential equation solvers

Single order differential equations are largely very well understood and can be numerically solved with standard differential equation solvers [26]. On the other hand, accuracy, and time efficiency of coupled ODEs and PDEs are of importance in many scientific fields [71]. It is known that QCs in the fault-tolerant scenario can provide an advantage over classical methods [72–75]. For near-term devices and variational quantum algorithms, it is generally believed that some gain could exist in specific problems [15, 16, 19, 20, 75–78]. To apply QKs methods to these more challenging equations, it is relevant to have at least a partial understanding of the heuristics behind the methods. In this context, numerical benchmarking studies can provide valuable insights into the methods and relevant hyperparameters settings, for example, as recently done for variational QML in Ref. [38].

There are many questions regarding QK solvers for differential equations that overlap with the main challenges of variational quantum algorithms. For example: which encoding circuit could provide an advantage over classical methods? In what kind of problems (or equations)? What is the impact of the size of the Hilbert space (i.e. exponential concentration)? What is the role of the hyperparameters in such methods? Here, we provide a first attempt to answer the latter. We investigate the impact of the number of qubits and the weight of the initial boundary condition in the performance of the QK solvers by performing a small benchmarking study.

Our framework is depicted in Fig. 4.9, we test for seven first-order differential equations

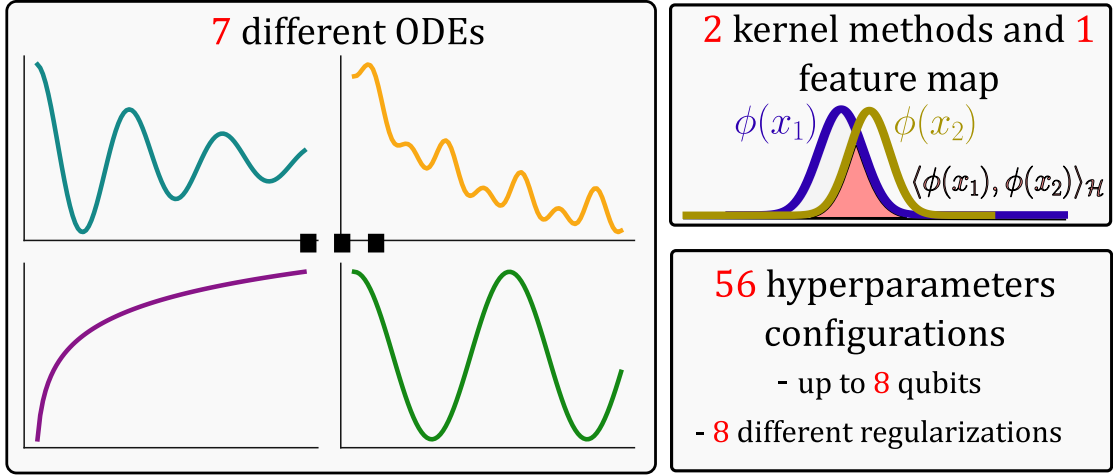


Figure 4.9.: Numerical setting of the hyperparameter study. We use the single-order differential equations specified in Table 4.1, each ODE is solved with FQKs and PQKs. The encoding circuit, Eq. 4.10, based on Chebyshev polynomials and controlled rotational gates is used. The number of qubits and the weight of the initial boundary condition are varied.

ODE Name	Equation
Log ODE	$\frac{df}{dx} - e^{-f(x)} = 0$
Polynomial with exponential growth	$2f(x) + 4\cos(x) - 8\sin(x) - \frac{df}{dx} = 0$
Oscillatory Decay 1	$\frac{df}{dx} + 8f(x)(0.1 + \tan(8x)) = 0$
Simple Cosine	$\frac{df}{dx} + \sin(x) = 0$
Nontrivial Dynamics	$\frac{df}{dx} - 4f(x) + 6f^2 - \sin(50x) - f(x)\cos(25x) + 0.5 = 0$
Logistic Equation	$f(x)(10 - f(x)) - \frac{df}{dx} = 0$
Oscillatory Decay 2	$\frac{df}{dx} + 20e^{-2x}\sin(20x) + 2f(x) = 0$

Table 4.1.: List of ODEs solved for the benchmarking study.

(see Table 4.1) and two QK methods, FQKs and PQKs. We change the number of qubits (2 to 8) and the weight of the initial condition boundary (From Eq. 2.47, $\eta \in [10^{-2}, 10^1]$). The data-encoding circuit is fixed and given by,

$$|\psi(x)\rangle = U(x)|0\rangle = \prod_i^L U_{\text{cheb}}(x)U_{\text{HEA}}(\theta)|0\rangle \quad (4.10)$$

where $U_{\text{cheb}}(x) = \bigotimes_j^n R_y(2j \arccos(x))$ and $U_{\text{HEA}}(\theta)$ corresponds to a sequence of rotations $R_z R_x R_z$ on each qubit, following entangling CNOT gates between the nearest

j	T_j	S_j
0	1	1
1	x	$2x$
2	$2x^2 - 1$	$4x^2 - 1$
3	$4x^3 - 3x$	$8x^3 - 4x$

Table 4.2.: First four Chebyshev Polynomials of the first and second kind. The rest can be derived by the recursion $T_{j+1}(x) = 2xT_j(x) - T_{j-1}(x)$ and $S_{j+1}(x) = 2xS_j(x) - S_{j-1}(x)$.

neighbors. The choice of this unitary was motivated by their use in the literature [15, 33], as the rotations within $U_{\text{cheb}}(x)$ generate the so called Chebyshev Polynomials. Expanding the R_y rotation, we find

$$R_y(2j \arccos(x)) = \exp\left(-i \frac{2j \arccos(x)}{2} Y\right), \quad (4.11)$$

$$= \cos(j \arccos(x)) \mathbb{1} - i \sin(j \arccos(x)) Y, \quad (4.12)$$

$$= T_j(x) \mathbb{1} + \sqrt{1 - x^2} S_{j-1}(x) XZ. \quad (4.13)$$

$$(4.14)$$

where X, Y, Z are the Pauli matrices, $\mathbb{1}$ is the identity matrix, and more importantly $T_j(x)$ and $S_j(x)$ are the Chebyshev Polynomials (Table 4.2) of the first and second kind, respectively [15]. These polynomials have simplified rules for combination and differentiation (see Ref. [79]). The main motivation for the choice of the Chebyshev Polynomials is that any smooth function can be approximated by a linear combination of these polynomials, i.e. $f(x) = \sum_j a_j T_j(x)$ [15]. By applying the gate in different qubits and layers, new polynomials are generated by the combination rules of the Chebyshev Polynomials. Therefore, the expressibility of the ansatz should increase.

We aim to statistically understand the effect of the hyperparameters in two main performance metrics. The mean squared error $\text{MSE}(f)$ between the predicted and the exact solution given by numerical standard methods and the total loss $\mathcal{L}_{\text{total}}$ which is minimized by the solver (i.e. Eq. 2.47). For the total loss, we also look at the individual contributions of the initial value loss $\mathcal{L}_{IV}$ and the differential equation $\mathcal{L}_f$.

4. Results

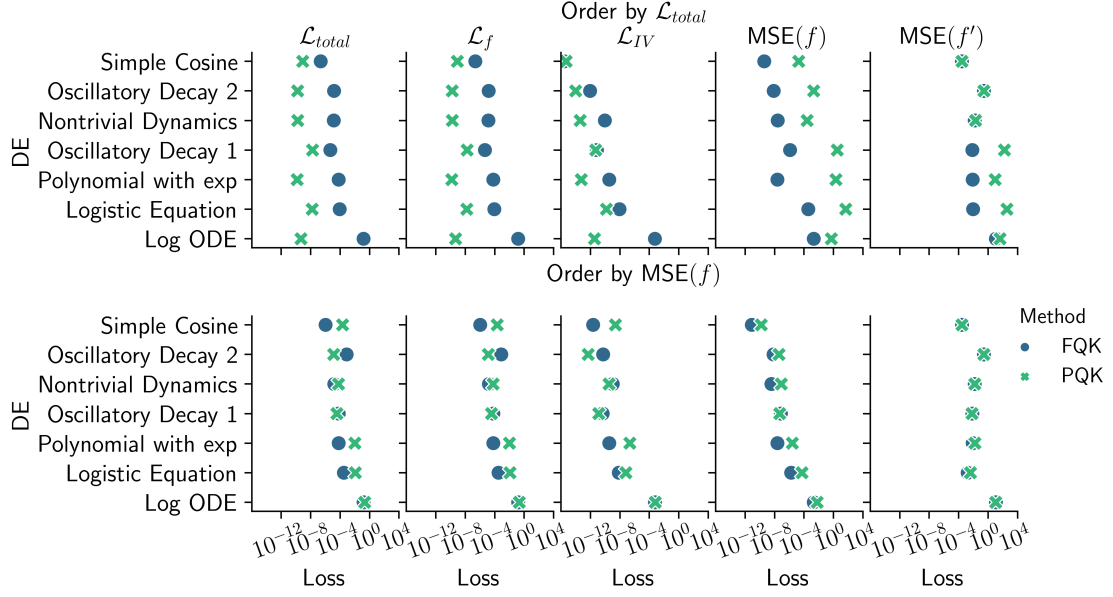


Figure 4.10.: Performance metrics of the best solver for each ODE for the hyperparameter study. Each row corresponds to different orderings, the first and second row correspond to ordering by the smallest total loss $\mathcal{L}_{total}$ and the smallest $MSE(f)$, respectively. The columns correspond to the different performance metrics, the total loss $\mathcal{L}_{total}$, the differential equation loss $\mathcal{L}_f$, the initial value loss $\mathcal{L}_{IV}$, the mean squared error of f , $MSE(f)$ and of f' , $MSE(f')$. We use green crosses (blue dots) for FQKs (PQKs).

4.3.1. Results

We compare the performance of the best solver for each ODE of the entire hyperparameter study. To judge each method, we look at the quantities described previously: $MSE(f)$, $\mathcal{L}_{total}$, $\mathcal{L}_f$, $\mathcal{L}_{IV}$. In Fig. 4.10, we plot the metrics of the best solver for each ODE when ordered by the smallest $\mathcal{L}_{total}$ (Fig. 4.10 - first row) and by the smallest $MSE(f)$ (Fig. 4.10 - second row).

Ordering by the smallest total loss $\mathcal{L}_{total}$ (Fig. 4.10 - first row), consistently shows that the PQKs have smaller losses than the FQKs. However, this is not reflected in the $MSE(f)$ measure, as FQKs have smaller errors. This hints that the best PQKs in regards to $\mathcal{L}_f$, are overfitting the differential equation loss. On the other hand, according to the MSE ordering (Fig. 4.10 - second row), the optimal FQKs always find better or equal solution than the optimal PQKs (in all of the errors measures). This is observed

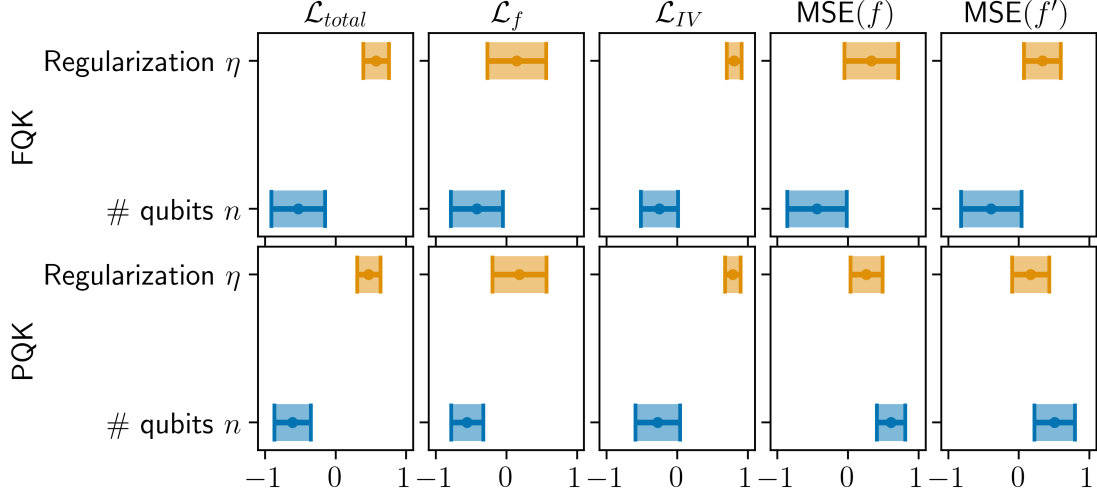


Figure 4.11.: Spearman correlation of the hyperparameter with the error metrics and losses averaged over the seven ODEs solved. From left to right, different performance metrics: the total loss $\mathcal{L}_{total}$, the differential equation loss $\mathcal{L}_f$, the initial value loss $\mathcal{L}_{IV}$, the mean squared error of f , $MSE(f)$ and of f' , $MSE(f')$. First/Second row corresponds to the correlations for FQKs/PQKs.

in all the ODEs, except for the Oscillatory Decay 2. In this case, although the optimal FQK has a smaller MSE, the total loss is smaller in the optimal PQK. This analysis was done for the best solver of each ODE. In the following paragraph we discuss which hyperparameter combination lead to small errors.

In Fig. 4.11, we numerically determined the Spearman correlation of the number of qubits and the regularization parameter with the corresponding performance metrics for FQKs and PQKs. With regards to the losses, both methods have very similar behavior. Generally, increasing the number of qubits and decreasing the regularization parameter η reduces the magnitude of the losses. The main difference between methods is the behavior of the metric $MSE(f)$. Surprisingly, for PQKs, as the number of qubits increases, the MSE increases. This is the opposite behavior of the FQK, and is discussed below.

Another way of understanding the correlations provided in Fig. 4.11 is to rank them. This way, it is possible to determine which hyperparameter gives most frequently the optimal answer. For example, how many times does $\eta = 0.027$ gives the smallest $MSE(f)$? These type of rankings provide a visual description of the spread of the data

and the empirical meaning of the correlations.

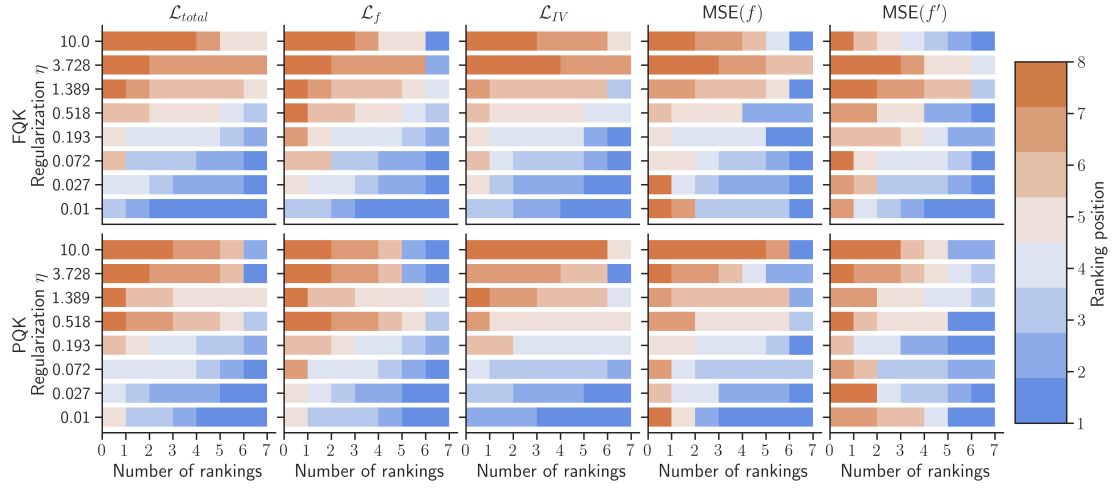
This analysis is shown in Fig. 4.12A (Fig. 4.12B). The solvers are ordered by the smallest $\text{MSE}(f)$, then, for every simulated η (n) the best solution available with respect to the number of qubits (regularization parameters) is selected. We report the number of times a method was ranked the i th best method. The ranking captures the same correlations of Fig. 4.11, which we discuss next.

As the number of qubits increases, we expect that the expressibility of the function ansatz produced by Eq. 4.10 increases. This is consistent with the numerical experiments for the FQKs: as the number of qubits increases, the solutions have a smaller MSE. On the other hand, PQKs, show contrary behavior, with increasing number of qubits, the MSE increases but the overall loss decreases. One possible explanation is that the model is overfitting $\mathcal{L}_f$, and that the regularization parameter η is not enough to prevent this. This would also explain why the solutions with the smallest $\mathcal{L}_f$, in the first row of Fig. 4.10, correspond to PQKs but always have smaller $\text{MSE}(f)$ than FQKs. One possible approach to understand this would be to test the trained models on a slightly different domain, similar to a test set.

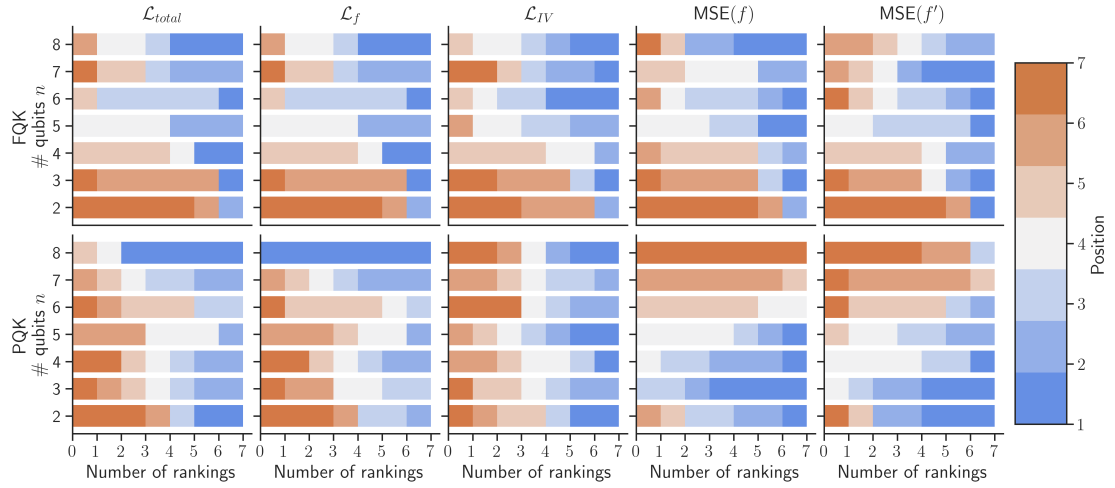
4.3.2. Discussion

The hyperparameter study was done in a very limited experimental setting. We only investigated seven ODEs and two QK methods. One of those methods, PQKs, have not been previously reported in the literature for ODE solvers. At first glance, we conclude that FQK solvers are better than PQK solvers. The number of qubits is the most important hyperparameter, and increasing it generally improves the performance of the FQK solver. On the other hand, we report hints that the PQK solver overfits the differential equation, as small total losses are achieved but the MSE does not correlate. As next steps further experiments with different test sets, could provide more insights into the behavior of the solvers. Furthermore, instead of looking at simple ODEs, experiments with PDEs and coupled ODEs that are classically challenging, could provide interesting points of view when comparing to classical PINNs.

4. Results



(A) Role of the regularization parameter η



(B) Role of the number of qubits n

Figure 4.12.: Number of times a method was ranked the i th best method for the $MSE(f)$ metric. Each i th position has an associated color given by the colorbar. Blue/Red corresponds to the best/worst method. The rest of the metrics correspond to the ranking given the $MSE(f)$ ordering. From left to right, different performance metrics: the total loss $\mathcal{L}_{total}$, the differential equation loss $\mathcal{L}_f$, the initial value loss $\mathcal{L}_{IV}$, the mean squared error of f , $MSE(f)$ and of f' , $MSE(f')$. From top to bottom: the rankings are given for different regularization parameter η (A) and number of qubits n (B).

5. Conclusion

Understanding the limitations and capabilities of Quantum Computers is a key point in their development. This is especially true for QML research, where many theoretical and applied challenges must be overcome to reach a state of quantum utility. From this perspective, understanding the practical inner working of methods is crucial. In this thesis, we explored many theoretical tools of QK methods in real examples. Furthermore, we implemented a differential equation solver based on variational quantum methods. The finding of the two research direction, are summarized in the following paragraphs.

First, multiplying the inputs of a QK by a small constant factor, known as bandwidth tuning, has been shown to improve the performance of QKs in classification tasks. In this thesis, we extensively studied and presented numerical experiments demonstrating how QKs become classically tractable. Using a matrix distance measure based on the Frobenius norm as well as the geometric difference, we revealed, that as the size of the scaling factor decreases, the QKs begin to resemble classical polynomial kernels. Furthermore, in the datasets studied we report that the ideal bandwidth, for large dimensions, is always in the regime where the QK shares the generalization properties, kernel variance and largest eigenvalue, with classical kernels. Additionally, we introduce an analytical model that reproduces how the kernel variance scales with the bandwidth and number of qubits for FQKs. Overall, bandwidth tuning will likely not provide any generalization advantage in these cases.

With this, we substantially contributed to a better understanding of the implications and connections of many methods in QK research. We utilized many of these techniques, which were mostly introduced from a complexity theoretic perspective, to understand and assess the performance of QKs applied to realistic scenarios. These are exponential concentration [42], bandwidth tuning [12–14, 52], generalization error [10, 13, 43, 55], kernel task-model alignment [13, 43, 55], and inductive bias in terms of kernel eigenvalue spectrum analysis [13, 43, 55].

Second, we implemented a single variable differential equation solver with QNNs in the Python library sQUlearn, and the fundamental building blocks for QKs solvers. The QNN implementation is ready-to-use and user-friendly in a way that users without

QML knowledge can directly work with it. Furthermore, it paves the way for implementations for PDEs and coupled ODEs solvers in the package. Additionally, as a small exploratory proof-of-concept, we performed a small hyperparameter study on the QKs. Here, we report for the first time, benchmarks with PQKs as original investigations were limited to FQKs [33] or QNNs [15, 19]. Although our results are in a small limited setting, they suggested that FQKs show smaller MSE than PQKs and that PQKs can suffer from overfitting more than FQKs.

As a final outlook, these two topics are not disconnected. Besides both studying QML methods, bandwidth tuning is a linking point. In the Fourier picture, bandwidth tuning can be seen as frequency tuning, i.e. the bandwidth parameter corresponds to a frequency parameter. For the first research direction, using moderate bandwidths as a frequency tuning parameter, might be an interesting direction to find problems where the periodic nature of QKs could have a generalization advantage over classical kernels. Similarly, exploring this connection for the quantum differential equations solvers might be a promising path. For instance, investigating if one can make use of similar techniques as bandwidth tuning to include frequency information in cases where the quantum models do not have native access to the exact frequency of a solution, may be a promising next step.

List of Figures

1.1. Outline of the thesis, four reading paths are suggested. In-between sections are omitted.	2
2.1. A Classification problem in a two-dimensional space is transformed to a three-dimensional space using a kernel function. The decision boundary becomes linear in the higher-dimensional space.	5
2.2. Schematic representation of the largest margin $2/\ w\ $ separating hyperplane in a two-dimensional space. The closest points to the hyperplane, with dotted lines, are the support vectors. Based on Ref. [25].	8
2.3. A visual representation of the difference between classical and QKs. In a FQK, points are mapped from a classical space to a quantum space through a density matrix, and the inner product corresponds to the fidelity between the two encoded states. PQKs represent an in-between approach, where the quantum state is projected to a classical space and is classically manipulated by an envelope function. Inspired by Fig. 2 from Ref. [10]	11
2.4. Schematic representation of circuits used to compute the most popular approaches in QML. FQKs and PQKs compare two points, while QNNs are a general ansatz for a function. Inspired by Fig. 1 from Ref. [37] . . .	12
2.5. Scheme representing the optimization process to solve a DE using a QNN (A) and a QK (B). The quantum circuit is parameterized by a QNN or a QK. The functions are evaluated on a quantum computer and optimized classically. Inspired by Fig. 1 from Ref. [15].	16

3.1. A. Pre-scaling the inputs that enter the kernel method, corresponds to using only a section of the full Hilbert space as defined by the data-encoding unitary. This results in FQK matrices with good spectral properties and well alignment to the learning problem [13] (adapted from Fig. 1 of Ref. [13]). B. Ref. [12] reported increases in the distance between the kernel and the identity matrix depending on the bandwidth for FQKs. C. Ref. [14] reported decreases with dimensionality in the geometrical difference of FQKs bandwidth-tuned kernels. D. Exponential concentration is reflected in the rank of a kernel matrix, for example, the rank of the identity matrix is given by its number of columns. A large rank corresponds to a flat eigenvalue spectrum [13]. E. The model complexity s_k of FQK is upper bounded by the rank of the kernel matrix [10]. F. Eigenvalues cannot be flat, and target functions must be aligned for a model to generalize well [13, 55]. Furthermore, the maximum eigenvalue has to be sufficiently large with respect to the number of data points for the model to generalize [43] (see Section 3.2). G. Ref. [12] and Refs. [52, 54] reported a numerical increase in performance for bandwidth-tuned kernels in FQKs and PQKs, respectively. H. Ref. [42] proved that for shot simulations, the kernel matrix of a FQK is independent of the input data. I. Geometrical difference compares the model complexity of two methods, the model complexity bounds the generalization error [10]. Opposite correlation between performance and geometrical difference is reported in Ref. [52]. 3.2. Upper panel: Schematic diagram of the projections $a_l = \overline{w}_{l'} \sqrt{\eta_{l'}}$ of two different target functions (blue/red) on the eigenfunctions spanned by a kernel. Here, only four eigenfunctions are drawn. Lower panel: Reproduction of the results of Fig. 1 of Ref. [55], task-model alignment and eigenvalues of two (quantum) kernels with good and bad generalization properties. 3.3. Spectral Bias and Task-Model Alignment of an RBF kernel ($\gamma = 0.5$). Classification of the make_moons dataset [65] and MNIST [66] digits 0 and 1. From left to right: (A) eigenvalues η_l and (B) task-model alignment C_l as a function of the index l, (C) theoretical (solid lines) and experimental generalization (scattered points) error from Eq. 3.8 and (D) accuracy as a function of the number of training points P. Because the task-model alignment of the MNIST01 simulations rises slower than in make_moons, more samples are needed to learn the target function, resulting in smaller accuracy. Inspired by Fig. 1 of Ref. [55]	20 25 25
---	-------------------------------

3.4. Flowchart to identify potential quantum advantage in kernel methods. Adapted from Fig. 1B of Ref. [10]	28
3.5. Cartoon representing the effect of rescaling the size of the data points in the feature space of the kernel. Shaydulin & Wild [12] reported that $0 \leq c \leq 1$ improves model accuracy.	29
3.6. Effect of bandwidth tuning in a toy dataset, there is an optimal value of c that minimizes the generalization error. From left to right: E_g as a function of bandwidth c , eigenvalue spectra η_l and task-model alignment C_l for two different bandwidths. Inspired by Fig. 1 of Ref. [12] and Fig. 2 of Ref. [13]. Numerical Setting: data-encoding unitary of Eq. 2.36 with $n = 9$ qubits and 3 re-uploading layers, 375 training and 225 testing points. Learning problem with a noisy target function $\hat{f}(x) = e^{-\ x\ ^2/9^2} + 0.05\epsilon$, ϵ corresponds to uniform sampled noise between 0 and 1. Each training vector is sampled from a uniform distribution $p(x \in [-\pi, \pi]^{\times 9}) = 1/2\pi$	30
4.1. Three-dimensional plot of the kernel matrix based on the IQP encoding circuit, Eq. 3.28 (upper panel) and a classical polynomial model, Eq. 4.1, of degree $d = 3$ (lower panel) for different rescaling values c . The first kernel entry x is fixed at (0.23, 1.47) and the other entries are uniformly sampled for $y \in \mathbb{R}^2$. The color scale and the height of the surface correspond to the kernel value.	35
4.2. Box plots of important quantities as a function of dimensionality, blue/yellow corresponds to the quantum/classical model and gray corresponds to the geometric difference $g(K_C, K_Q)$ and the Frobenius distance $F(K_C, K_Q)$ measures which depend on kernel matrices of both models. The boxes are constructed from the models with optimal bandwidths c^* , yielding the highest roc auc scores for each seed. Figures from left to right depict: roc auc score, largest eigenvalue $\eta_{\max}$, optimal bandwidth c^* , geometrical difference $g(K_C, K_Q)$ and Frobenius distance $F(K_C, K_Q)$. From top to bottom different models: (A) IQP FQK (plasticc), (B) IQP PQK (plasticc) and (C) HamEvol FQK (kMNIST28). The red dashed line for the geometrical difference corresponds to $\sqrt{N}$, the limit where no quantum advantage is possible. For increasing dimensions, the geometric difference and the Frobenius distance between the classical and quantum models decreases. This is reflected in similar performance and largest eigenvalue.	39

- 4.3. Eigenvalues η_l of the kernel matrices. Different colors (cf. legend) correspond to different bandwidth tuning parameters and the number of dimensions is fixed to $n = 16$. The median optimal bandwidth is represented by " c^* " (light blue solid lines). The lines correspond to the median eigenvalues over the different seeds used. Each subfigure corresponds to a different model. From left to right: (A) poly_2 ($c^* = 0.062$), (B) IQP FQK ($c^* = 0.062$) and (C) IQP PQK ($c^* = 0.032$) for the plasticc dataset, and, (D) poly_2 ($c^* = 0.016$) and (E) HamEvol FQK ($c^* = 0.12$) for the kMNIST28 dataset. 40
- 4.4. Task-model alignment C_l of the kernel matrices. Different colors (cf. legend) correspond to different bandwidth tuning parameters and the number of dimensions is fixed to $n = 16$. Each subfigure corresponds to a different model. The median optimal bandwidth is represented by " c^* " (light blue solid lines). The lines correspond to the median C_l over the different seeds used. From left to right: (A) poly_2 ($c^* = 0.062$), (B) IQP FQK ($c^* = 0.062$) and (C) IQP PQK ($c^* = 0.032$) for the plasticc dataset, and, (D) poly_2 ($c^* = 0.016$) and (E) HamEvol FQK ($c^* = 0.12$) for the kMNIST28 dataset. 41
- 4.5. Relevant measures as a function of bandwidth averaged over the different simulated seeds. Different colors correspond to different dimensionality, the darker the color the larger the number of dimensions. The shaded area corresponds to the standard deviation of the models. Dashed lines represent the classical models (either poly_2 or poly_4), whereas solid lines represent the quantum models. From top to bottom: (A) IQK (FQK) and plasticc, (B) IQK (PQK) and plasticc, (C) HamEvol (FQK) and kMNIST28. From left to right: variance of the kernel entries $\text{Var}[K]$, largest eigenvalue $\eta_{\max}$, Frobenius distance $F(K_C, K_Q)$ and geometrical difference $g(K_C, K_Q)$. The horizontal dot-dashed line in the geometrical difference plots corresponds to $\sqrt{N}$, the limit where no quantum advantage is possible. The variance and the largest eigenvalue of the quantum models exhibit three distinct regimes: small, intermediate and large bandwidths. In the small bandwidth regime, the classical and quantum models share the same behavior. The optimal bandwidths primarily occur within the small bandwidth regime and during the shift from small to intermediate bandwidths 44

- 4.6. Variance of the kernel as a function of bandwidth for the separable R_x encoding circuit based FQKs for two toy datasets. The first column, depicts the three regimes of the variance for the uniform distribution, with $n = 6$. The second/third column corresponds to the uniform/cosine distribution simulations. Three curves are shown for different number of dimensions $n = 1, 3, 5$. The numerics (blue solid lines) were obtained by sampling 1200 points from the corresponding distribution and calculating the variance of the kernel entries. For the analytical plots (orange dashed lines), we used the exact equations Eq. A.36 and Eq. 4.5, respectively. In dotted lines, we show the limiting behavior calculated with Eq. 4.6 and Eq. 4.7 for the uniform distribution, and Eq. A.40 and Eq. A.41 for the raised cosine distribution. As shown in Fig. 4.5 (first column), the experiments for real datasets using QKs based on encoding circuits with entangling gates show the same three regimes and limiting behaviors. . 47
- 4.7. Frobenius distances according to Eq. 4.2 heatmap between kernel matrices of the two datasets. Each block represents the distance between the quantum (on the x -axis) and classical (on the y -axis) kernel matrices. For each dataset two different bandwidths are shown. The c^* column corresponds to the matrices with optimal bandwidth (highest roc auc score), and the second column has $c = 0.935$. As the number of qubits increases, the distance in the c^* column decreases whereas in the other column continues to be large. The distances in `kMNIST28` (left panel) are smaller than in `plasticc` (right panel). The RBF kernel is an exception as its values do not explode for c close to 1, see main text for a detailed discussion. 48
- 4.8. Geometric difference according to Eq. 3.20 between kernel matrices of the two datasets. Each block represents the difference between the quantum (on the x -axis) and classical (on the y -axis) kernel matrices. For each dataset two different bandwidths are shown. The c^* column corresponds to the matrices with optimal bandwidth (highest roc auc score), and the second column has $c = 0.935$. As the number of qubits increases, the geometric difference in the c^* column decreases whereas in the other column continues to be large. The geometric differences in `kMNIST28` (left panel) are smaller than in `plasticc` (right panel). The RBF kernel is an exception as it has a small geometric difference for almost all the blocks. 50

4.9. Numerical setting of the hyperparameter study. We use the single-order differential equations specified in Table 4.1, each ODE is solved with FQKs and PQKs. The encoding circuit, Eq. 4.10, based on Chebyshev polynomials and controled rotational gates is used. The number of qubits and the weight of the initial boundary condition are varied.	55
4.10. Performance metrics of the best solver for each ODE for the hyperparameter study. Each row corresponds to different orderings, the first and second row correspond to ordering by the smallest total loss $\mathcal{L}_{total}$ and the smallest $MSE(f)$, respectively. The columns correspond to the different performance metrics, the total loss $\mathcal{L}_{total}$, the differential equation loss $\mathcal{L}_f$, the initial value loss $\mathcal{L}_{IV}$, the mean squared error of f , $MSE(f)$ and of f' , $MSE(f')$. We use green crosses (blue dots) for FQKs (PQKs). .	57
4.11. Spearman correlation of the hyperparameter with the error metrics and losses averaged over the seven ODEs solved. From left to right, different performance metrics: the total loss $\mathcal{L}_{total}$, the differential equation loss $\mathcal{L}_f$, the initial value loss $\mathcal{L}_{IV}$, the mean squared error of f , $MSE(f)$ and of f' , $MSE(f')$. First/Second row corresponds to the correlations for FQKs/PQKs.	58
4.12. Number of times a method was ranked the i th best method for the $MSE(f)$ metric. Each i th position has an associated color given by the colorbar. Blue/Red corresponds to the best/worst method. The rest of the metrics correspond to the ranking given the $MSE(f)$ ordering. From left to right, different performance metrics: the total loss $\mathcal{L}_{total}$, the differential equation loss $\mathcal{L}_f$, the initial value loss $\mathcal{L}_{IV}$, the mean squared error of f , $MSE(f)$ and of f' , $MSE(f')$. From top to bottom: the rankings are given for different regularization parameter η (A) and number of qubits n (B).	60
A.1. IQP encoding circuit as used by Ref. [12] in repository [80].	81
A.2. 1D Heisenberg interaction encoding feature map as described by Ref. [14] based on previous implementation in repository [80].	81

- A.3. Repetition of Fig. 4.2 for the remaining datasets. Box plots of important quantities as a function of dimensionality, blue/yellow corresponds to the quantum/classical model and gray corresponds to the geometric difference $g(K_C, K_Q)$ and the Frobenius distance $F(K_C, K_Q)$ measures which depend on kernel matrices of both models. The boxes are constructed from the models with optimal bandwidths c^* , yielding the highest roc auc scores for each seed. Figures from left to right depict: roc auc score, largest eigenvalue $\eta_{\max}$, optimal bandwidth c^* , geometrical difference $g(K_C, K_Q)$ and Frobenius distance $F(K_C, K_Q)$. From top to bottom different models: (A) IQP FQK (kMNIST28), (B) IQP PQK (kMNIST28) and (C) HamEvol FQK (plasticc). The red dashed line for the geometrical difference corresponds to $\sqrt{N}$, the limit where no quantum advantage is possible. For increasing dimensions, the geometric difference and the Frobenius distance between the classical and quantum models decreases. This is reflected in similar performance and largest eigenvalue. 83
- A.4. Repetition of 4.5 for the remaining datasets. Relevant measures as a function of bandwidth averaged over the different simulated seeds. Different colors correspond to different dimensionality, the darker the color the larger the number of dimensions. The shaded area corresponds to the standard deviation of the models. Dashed lines represent the classical models (here poly_2), whereas solid lines represent the quantum models. From top to bottom: (A) IQK (FQK) and kMNIST28, (B) IQK (PQK) and kMNIST28, (C) HamEvol (FQK) and plasticc. From left to right: variance of the kernel entries $\text{Var}[K]$, largest eigenvalue $\eta_{\max}$, Frobenius distance $F(K_C, K_Q)$ and geometrical difference $g(K_C, K_Q)$. The horizontal dot-dashed line in the geometrical difference plots corresponds to $\sqrt{N}$, the limit where no quantum advantage is possible. 84
- A.5. Eigenvalue spectra for increasing dimensionality classical and quantum models used for the plasticc experiments. Extension of Fig. 4.3 for the remaining kernels. From left to right, different kernel models: poly_1, poly_2, poly_3, poly_4, RBF, IQP FQK, IQP PQK and HamEvol FQK. From top to bottom, different dimensions: 2, 4, 8, 10, 16. 85
- A.6. Eigenvalue spectra for increasing dimensionality classical and quantum models used for the kMNIST28 experiments.. Extension of Fig. 4.3 for the remaining kernels. From left to right, different kernel models: poly_1, poly_2, poly_3, poly_4, RBF, IQP FQK, IQP PQK and HamEvol FQK. From top to bottom, different dimensions: 2, 4, 8, 10, 16. 86

A.7. Task-model alignment for increasing dimensionality classical and quantum models used for the <code>plasticc</code> experiments. Extension of Fig. 4.4 for the remaining kernels. From left to right, different kernel models: <code>poly_1</code> , <code>poly_2</code> , <code>poly_3</code> , <code>poly_4</code> , RBF, IQP FQK, IQP PQK and HamEvol FQK. From top to bottom, different dimensions: 2, 4, 8, 10, 16.	87
A.8. Task-model alignment for increasing dimensionality classical and quantum models used for the <code>kmnist28</code> experiments. Extension of Fig. 4.4 for the remaining kernels. From left to right, different kernel models: <code>poly_1</code> , <code>poly_2</code> , <code>poly_3</code> , <code>poly_4</code> , RBF, IQP FQK, IQP PQK and HamEvol FQK. From top to bottom, different dimensions: 2, 4, 8, 10, 16.	88
A.9. Raised cosine $p(x) = 1/2\pi (1 + \cos(x))$ and uniform distributions $p(x) = 1$ for $n = 1$. As used for the derivation of Eq. A.27 and Eq. A.36. For the numerical experiments, 1200 points were sampled.	91
A.10. Overview of the main components of the sQUlearn QNN engine including the differential equation solver contained in ODELoss. Adapted from Ref. [17]	94
A.11. sQUlearn QNN differential equation example	96

List of Tables

2.1. Some examples of kernel functions and their corresponding feature maps, for one-dimensional variables $x, y \in \mathbb{R}^1$	8
2.2. Activation Functions for Neural Networks.	9
3.1. Summary of the main requirements for kernel generalization and their specific implications for FQKs.	32
4.1. List of ODEs solved for the benchmarking study.	55
4.2. First four Chebyshev Polynomials of the first and second kind. The rest can be derived by the recursion $T_{j+1}(x) = 2xT_j(x) - T_{j-1}(x)$ and $S_{j+1}(x) = 2xS_j(x) - S_{j-1}(x)$	56

Bibliography

1. Shor, P. W. Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. *SIAM Journal on Computing* **26**, 1484–1509 (1997).
2. Nielsen, M. A. & Chuang, I. L. *Quantum computation and quantum information* 10th anniversary ed. (Cambridge University Press, Cambridge and New York, 2010).
3. Pelucchi, E. *et al.* The potential and global outlook of integrated photonics for quantum technologies. *Nature Reviews Physics* **4**, 194–208 (2022).
4. Schäfer, F., Fukuhara, T., Sugawa, S., Takasu, Y. & Takahashi, Y. Tools for quantum simulation with ultracold atoms in optical lattices. *Nature Reviews Physics* **2**, 411–425 (2020).
5. Chatterjee, A. *et al.* Semiconductor qubits in practice. *Nature Reviews Physics* **3**, 157–177 (2021).
6. Biamonte, J. *et al.* Quantum machine learning. *Nature* **549**, 195–202 (2017).
7. Schuld, M. & Petruccione, F. *Machine Learning with Quantum Computers* 2nd ed. 2021 (Springer International Publishing and Springer, Cham, 2021).
8. Schuld, M. & Killoran, N. Is Quantum Advantage the Right Goal for Quantum Machine Learning? *PRX Quantum* **3**, 030101 (2022).
9. Cerezo, M. *et al.* Variational quantum algorithms. *Nature Reviews Physics* **3**, 625–644 (2021).
10. Huang, H.-Y. *et al.* Power of data in quantum machine learning. *Nature Communications* **12**, 2631 (2021).
11. Havlíček, V. *et al.* Supervised learning with quantum-enhanced feature spaces. *Nature* **567**, 209–212 (2019).
12. Shaydulin, R. & Wild, S. M. Importance of kernel bandwidth in quantum machine learning. *Physical Review A* **106**, 042407 (2022).
13. Canatar, A., Peters, E., Pehlevan, C., Wild, S. M. & Shaydulin, R. Bandwidth Enables Generalization in Quantum Kernel Models. *Transactions on Machine Learning Research* (2023).

14. Slattery, L. *et al.* Numerical evidence against advantage with quantum fidelity kernels on classical data. *Physical Review A* **107** (2023).
15. Kyriienko, O., Paine, A. E. & Elfving, V. E. Solving nonlinear differential equations with differentiable quantum circuits. *Physical Review A* **103** (2021).
16. Paine, A. E., Elfving, V. E. & Kyriienko, O. Quantum kernel methods for solving regression problems and differential equations. *Physical Review A* **107** (2023).
17. Kreplin, D. A. *et al.* sQUlearn – A Python Library for Quantum Machine Learning. 2024. arXiv: 2311.08990 [quant-ph].
18. Knudsen, M. & Mendl, C. B. Solving Differential Equations via Continuous-Variable Quantum Computers. 2020. arXiv: 2012.12220 [quant-ph].
19. Setty, A., Abdusalamov, R. & Itskov, M. Physics-Informed Quantum Machine Learning for Solving Partial Differential Equations. 2023. arXiv: 2312.09215 [quant-ph].
20. Jaderberg, B., Gentile, A. A., Berrada, Y. A., Shishenina, E. & Elfving, V. E. Let quantum neural networks choose their own frequencies. *Physical Review A* **109**, 042421 (2024).
21. Han Veiga, M. & Ged, F. G. *The Mathematics of Machine Learning: Lectures on Supervised Methods and Beyond* (De Gruyter, Boston, 2024).
22. Murphy, K. P. *Machine learning: A probabilistic perspective* (MIT Press, Cambridge MA, 2012).
23. Hastie, T., Tibshirani, R. & Friedman, J. H. *The elements of statistical learning: Data mining, inference, and prediction* 2nd ed. (Springer, New York NY, 2009).
24. Scholkopf, B. & Smola, A. J. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond* (MIT Press and Ebsco Publishing [distributor], Cambridge and Ipswich).
25. contributors, W. SVM margin — Wikipedia, the free encyclopedia. https://en.wikipedia.org/wiki/File:SVM_margin.png. 2018.
26. Zill, D. *A First Course in Differential Equations with Modeling Applications* (Cengage Learning, 2016).
27. Karniadakis, G. E. *et al.* Physics-informed machine learning. *Nature Reviews Physics* **3**, 422–440 (2021).
28. Cuomo, S. *et al.* Scientific Machine Learning through Physics-Informed Neural Networks: Where we are and What’s next. *CoRR* (2022).
29. Guan, W. *et al.* Quantum machine learning in high energy physics. *Machine Learning: Science and Technology* **2**, 011003 (2021).

30. Mitarai, K., Negoro, M., Kitagawa, M. & Fujii, K. Quantum circuit learning. *Physical Review A* **98** (2018).
31. Arrasmith, A., Holmes, Z., Cerezo, M. & Coles, P. J. Equivalence of quantum barren plateaus to cost concentration and narrow gorges. *Quantum Science and Technology* **7**, 045015 (2022).
32. Cerezo, M., Sone, A., Volkoff, T., Cincio, L. & Coles, P. J. Cost function dependent barren plateaus in shallow parametrized quantum circuits. *Nature Communications* **12**, 1791 (2021).
33. Paine, A. E., Elfving, V. E. & Kyriienko, O. Quantum kernel methods for solving regression problems and differential equations. 2023.
34. Henry, L.-P., Thabet, S., Dalyac, C. & Henriët, L. Quantum evolution kernel: Machine learning on graphs with programmable arrays of qubits. *Physical Review A* **104** (2021).
35. Hubregtsen, T. *et al.* Training quantum embedding kernels on near-term quantum computers. *Physical Review A* **106** (2022).
36. Schuld, M. Supervised quantum machine learning models are kernel methods. 2021. arXiv: 2101.11020 [quant-ph].
37. Jerbi, S. *et al.* Quantum machine learning beyond kernel methods. *Nature Communications* **14**, 517 (2023).
38. Bowles, J., Ahmed, S. & Schuld, M. Better than classical? The subtle art of benchmarking quantum machine learning models. arXiv: 2403.07059 [quant-ph].
39. Liu, Y., Arunachalam, S. & Temme, K. A rigorous and robust quantum speed-up in supervised machine learning. *Nature Physics* **17**, 1013–1017 (2021).
40. Schuld, M. & Killoran, N. Quantum Machine Learning in Feature Hilbert Spaces. *Physical Review Letters* **122**, 040504 (2019).
41. Havlíček, V. *et al.* Supervised learning with quantum-enhanced feature spaces. *Nature* **567**, 209–212 (2019).
42. Thanasilp, S., Wang, S., Cerezo, M. & Holmes, Z. Exponential concentration in quantum kernel methods. *Nature Communications* **15**, 5200 (2024).
43. Kübler, J. M., Buchholz, S. & Schölkopf, B. The Inductive Bias of Quantum Kernels. 2021. arXiv: 2106.03747 [quant-ph].
44. Cerezo, M. *et al.* Does provable absence of barren plateaus imply classical simulability? Or, why we need to rethink variational quantum computing. 2024. arXiv: 2312.09121 [quant-ph].

45. Schuld, M., Sweke, R. & Meyer, J. J. Effect of data-encoding on the expressive power of variational quantum-machine-learning models. *Physical Review A* **103** (2021).
46. Crooks, G. E. Gradients of parameterized quantum gates using the parameter-shift rule and gate decomposition. 2019. arXiv: 1905.13311 [quant-ph].
47. Schuld, M., Bergholm, V., Gogolin, C., Izaac, J. & Killoran, N. Evaluating analytic gradients on quantum hardware. *Physical Review A* **99** (2019).
48. Hubregtsen, T., Wilde, F., Qasim, S. & Eisert, J. Single-component gradient rules for variational quantum algorithms. *Quantum Science and Technology* **7**, 035008 (2022).
49. Wierichs, D., Izaac, J., Wang, C. & Lin, C. Y.-Y. General parameter-shift rules for quantum gradients. *Quantum* **6**, 677 (2022).
50. Bergholm, V. *et al.* PennyLane: Automatic differentiation of hybrid quantum-classical computations. 2022. arXiv: 1811.04968 [quant-ph].
51. Javadi-Abhari, A. *et al.* Quantum computing with Qiskit. 2024. arXiv: 2405.08810 [quant-ph].
52. Egger, S., Sakhnenko, A. & Lorenz, J. M. A hyperparameter study for quantum kernel methods. *Quantum Machine Intelligence* **6**, 1–18 (2024).
53. Suzuki, Y., Kawaguchi, H. & Yamamoto, N. Quantum Fisher kernel for mitigating the vanishing similarity issue. 2022. arXiv: 2210.16581 [quant-ph].
54. Gan, B. Y., Leykam, D. & Thanasilp, S. A Unified Framework for Trace-induced Quantum Kernels. 2023. arXiv: 2311.13552 [quant-ph].
55. Canatar, A., Bordelon, B. & Pehlevan, C. Spectral bias and task-model alignment explain generalization in kernel regression and infinitely wide neural networks. *Nature Communications* **12**, 2914 (2021).
56. Abbas, A. *et al.* The power of quantum neural networks. *Nature computational science* **1**, 403–409 (2021).
57. Uvarov, A. V. & Biamonte, J. D. On barren plateaus and cost function locality in variational quantum algorithms. *Journal of Physics A: Mathematical and Theoretical* **54**, 245301 (2021).
58. Pesah, A. *et al.* Absence of Barren Plateaus in Quantum Convolutional Neural Networks. *Physical Review X* **11** (2021).
59. Ortiz Marrero, C., Kieferová, M. & Wiebe, N. Entanglement-Induced Barren Plateaus. *PRX Quantum* **2** (2021).

60. Cerezo, M. & Coles, P. J. Higher order derivatives of quantum neural networks with barren plateaus. *Quantum Science and Technology* **6**, 035006 (2021).
61. Holmes, Z., Sharma, K., Cerezo, M. & Coles, P. J. Connecting Ansatz Expressibility to Gradient Magnitudes and Barren Plateaus. *PRX Quantum* **3** (2022).
62. Olivia Di Matteo. Unitary designs. *PennyLane Demos* (2021).
63. Sim, S., Johnson, P. D. & Aspuru-Guzik, A. Expressibility and Entangling Capability of Parameterized Quantum Circuits for Hybrid Quantum-Classical Algorithms. *Advanced Quantum Technologies* **2**, 619 (2019).
64. Mézard, M., Parisi, G. & Virasoro, M. Á. *Spin Glass Theory and Beyond: An Introduction to the Replica Method and Its Applications* (2024).
65. Pedregosa, F. *et al.* Scikit-learn: Machine Learning in Python. 2018. arXiv: 1201.0490 [cs.LG].
66. Deng, L. The MNIST Database of Handwritten Digit Images for Machine Learning Research. *IEEE Signal Processing Magazine* **29**, 141–142 (2012).
67. Bremner, M. J., Montanaro, A. & Shepherd, D. J. Average-Case Complexity Versus Approximate Simulation of Commuting Quantum Computations. *Physical Review Letters* **117**, 080501 (2016).
68. team, P. The Photometric LSST Astronomical Time-series Classification Challenge (PLAsTiCC): Data set. 2018. arXiv: 1810.00001 [astro-ph.IM].
69. Clanuwat, T. *et al.* Deep Learning for Classical Japanese Literature. arXiv: 1812.01718 (2018).
70. Bermejo, P. *et al.* Quantum Convolutional Neural Networks are (Effectively) Classically Simulable. 2024. arXiv: 2408.12739 [quant-ph].
71. Städter, P., Schälte, Y., Schmiester, L., Hasenauer, J. & Stapor, P. L. Benchmarking of numerical integration methods for ODE models of biological systems. *Scientific Reports* **11**, 2696 (2021).
72. Harrow, A. W., Hassidim, A. & Lloyd, S. Quantum algorithm for linear systems of equations. *Physical Review Letters* **103**, 150502 (2009).
73. Babbush, R., Berry, D. W., Kothari, R., Somma, R. D. & Wiebe, N. Exponential Quantum Speedup in Simulating Coupled Classical Oscillators. *Physical Review X* **13** (2023).
74. Montanaro, A. & Pallister, S. Quantum algorithms and the finite element method. *Physical Review A* **93** (2016).

75. Mouton, L., Reiter, F., Chen, Y. & Rebentrost, P. Deep-learning-based quantum algorithms for solving nonlinear partial differential equations. *Physical Review A* **110**, 022612 (2024).
76. Oz, F., San, O. & Kara, K. An efficient quantum partial differential equation solver with chebyshev points. *Scientific Reports* **13**, 7767 (2023).
77. Zanger, B., Mendl, C. B., Schulz, M. & Schreiber, M. Quantum Algorithms for Solving Ordinary Differential Equations via Classical Integration Methods. *Quantum* **5**, 502 (2021).
78. Paine, A. E., Elfving, V. E. & Kyriienko, O. Physics-Informed Quantum Machine Learning: Solving nonlinear differential equations in latent spaces without costly grid evaluations. 2023. arXiv: 2308.01827 [quant-ph].
79. Mason, J. & Handscomb, D. *Chebyshev Polynomials* 1st (Chapman and Hall/CRC, 2002).
80. Shaydulin, R. & Wild, S. M. Importance of Kernel Bandwidth in Quantum Machine Learning. GitHub repository, <https://github.com/rsln-s/Importance-of-Kernel-Bandwidth-in-Quantum-Machine-Learning/tree/main>. Accessed: 2023-10-05. 2022.
81. Kingma, D. P. & Ba, J. Adam: A Method for Stochastic Optimization. 2017. arXiv: 1412.6980 [cs.LG].

A. Appendix

A.1. Matrix Inversion Lemma

From Ref. [22], Eq. 4.107:

Definition 7. *Matrix Inversion Lemma.* Consider the matrices E, F, H, G , where E and H are invertible. Then, the following identity holds:

$$(E + FH^{-1}G)^{-1}FH^{-1} = E^{-1}F(H + GE^{-1}F)^{-1} \quad (\text{A.1})$$

For the weights of the prediction from the kernel ridge regression, Eq. 2.10, take $E = \mathbb{I}_p$, $F = \Phi^T$, $H = \mathbb{I}_N$ and $G = \Phi\lambda^{-1}$. Then,

$$(\Phi^T\Phi + \lambda\mathbb{I}_p)^{-1}\Phi^T = (\Phi^T\Phi\lambda^{-1} + \mathbb{I}_p)^{-1}\Phi^T\lambda^{-1} \quad (\text{A.2})$$

$$= \mathbb{I}_p\Phi^T(\mathbb{I}_N + \Phi\Phi^T\lambda^{-1})^{-1}\lambda^{-1} \quad (\text{A.3})$$

$$= \Phi^T(\mathbb{I}_N\lambda + \Phi\Phi^T)^{-1} \quad (\text{A.4})$$

A.2. Theory of a Reproducing Kernel Hilbert Space

For an inner product to be a valid kernel, it must satisfy the following definition.

Definition 8. Kernel. Let $\mathcal{X}$ be a nonempty input set, a function $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ is a kernel if and only if it,

1. Is symmetric: $k(\mathbf{x}, \mathbf{x}') = k(\mathbf{x}', \mathbf{x})$.
2. Is positive semi-definite: for any set of points $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathcal{X}$ the Gramm matrix K with elements $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$ has all its eigenvalues non-negative.

Each kernel function k has an associated reproducing kernel Hilbert space $\mathcal{H}_k$.

Definition 9. *Reproducing Kernel Hilbert Space.* Let $\mathcal{X}$ be a nonempty input set and $\mathcal{H}_k$ a Hilbert space of functions $f : \mathcal{X} \rightarrow \mathbb{R}$. Then $\mathcal{H}_k$ is called a reproducing kernel Hilbert space

endowed with the dot product $\langle \cdot, \cdot \rangle$ if there exists a function $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ with the following properties:

1. For all $\mathbf{x} \in \mathcal{X}$, the reproducing property holds: $f(\mathbf{x}) = \langle f, k(\mathbf{x}, \cdot) \rangle$, this includes $f = k(\mathbf{x}', \cdot) \implies \langle f, k(\mathbf{x}, \cdot) \rangle = \langle k(\mathbf{x}', \cdot), k(\mathbf{x}, \cdot) \rangle = k(\mathbf{x}', \mathbf{x})$.
2. For all $\mathbf{x} \in \mathcal{X}$, the kernel $k(\mathbf{x}, \cdot)$ is in $\mathcal{H}_k$.

By combining different mappings, we can construct all the elements of the Reproducing Kernel Hilbert Space (RKHS).

$$f(\cdot) = \sum_{i=1}^n \alpha_i k(\mathbf{x}_i, \cdot) \quad (\text{A.5})$$

We can think of the possible f , as vector functions in an infinite-dimensional space. Evaluating a function f in a point $\mathbf{z}$, corresponds to the inner product $\langle f, k(\mathbf{z}, \cdot) \rangle = f(\mathbf{z}) = k(\mathbf{x}, \mathbf{z})$.

A.3. Mercer's Theorem

The continuous analogous to the kernel matrix is given by the kernel integral operator.

Definition 10. *Kernel Integral operator.* Let $\mathcal{X}$ be a nonempty input set, where $\mathbf{x} \in \mathcal{X}$ follows a probability distribution $p(\mathbf{x})$. Let k be a valid kernel function with an associated Hilbert space $\mathcal{H}_k$ of functions $f : \mathcal{X} \rightarrow \mathbb{R}$. The integral operator $T_k : \mathcal{L}_2(\mathcal{X}) \rightarrow \mathcal{L}_2(\mathcal{X})$ associated with the kernel k , is defined as,

$$T_k f(\mathbf{x}) = \int_{\mathcal{X}} k(\mathbf{x}, \mathbf{x}') f(\mathbf{x}') p(\mathbf{x}) d\mathbf{x} \quad (\text{A.6})$$

T_k is the continuous analogous of the Gramm matrix $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$.

A kernel can be decomposed into a sum of eigenfunctions and eigenvalues of the integral operator T_k . This is known as Mercer's theorem [24].

Theorem 5. *Mercer's Theorem.* Let T_k be the integral operator associated with the kernel k . There is an orthogonal basis of eigenfunctions $T_k \Phi_l = \eta_l \Phi_l$, that spans $\mathcal{L}_2(\mathcal{X})$, such that the kernel k can be written as,

$$k(\mathbf{x}, \mathbf{x}') = \sum_{l=1}^{\infty} \eta_l \Phi_l(\mathbf{x}) \phi_l(\mathbf{x}') \quad (\text{A.7})$$

where the eigenvalues λ_l are non-negative and the eigenfunctions ϕ_l are orthogonal.

As a consequence of Mercer's theorem, any function $\bar{f} \in \mathcal{H}_k$ in $\mathcal{L}_2(\mathcal{X})$ can be written as [13],

$$\bar{f}(\mathbf{x}) = \sum_{i=1}^{\infty} a_i \Phi_i(\mathbf{x}), \quad a_i = \langle \bar{f}, \Phi_i \rangle_{\mathcal{H}_k} \quad (\text{A.8})$$

For the finite dimensional case [24],

$$\begin{aligned}
k(\mathbf{x}_i, \mathbf{x}_j) &= K_{ij} \\
&= (\mathbf{\Phi} \mathbf{\Lambda} \mathbf{\Phi}^T)_{ij} \\
&= \sum_{t=1}^n \eta_t v_{ti} v_{tj} \\
&= \sum_{t=1}^n \eta_t \Phi_t(\mathbf{x}_i) \Phi_t(\mathbf{x}_j)
\end{aligned} \tag{A.9}$$

where $\mathbf{\Phi}$ is the matrix of eigenvectors and $\mathbf{\Lambda}$ is the diagonal matrix of eigenvalues η .

As the series of Eq. A.7 is uniformly convergent, even for infinite-dimensional feature spaces, there exist an n , in Eq. A.9, such that the $k(\mathbf{x}_i, \mathbf{x}_j)$ can be ϵ approximated by a finite sum (see Proposition 2.11 in Ref. [24]).

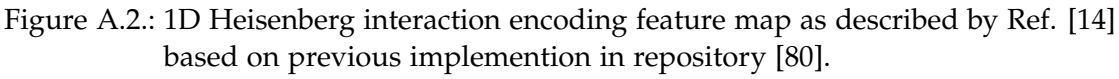
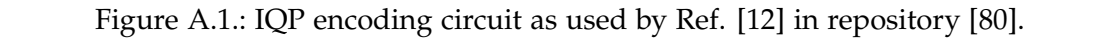
A.4. Technical details on numerical experiments of bandwidth-tuned kernels

We report our findings on numerical experiments of bandwidth-tuned kernels applied to the `plasticc` [68] and `kMNIST28` [69] datasets.

For `plasticc` we use the same dataset as provided by Ref. [12] and for `kMNIST28` we compare classes 2 and 8. The original dataset is transformed to an n -dimensional one using principal component analysis (PCA) and is preprocess by standardizing the features (mean 0, variance 1) and then performing PCA. For each experiment, 400 points are sampled from the original datasets using seeds 1 to 6, and split 80%/20% for training/testing. We use 35 rescaling factors $c \in [10^{-4}, 10^1]$ in a logarithmic scale.

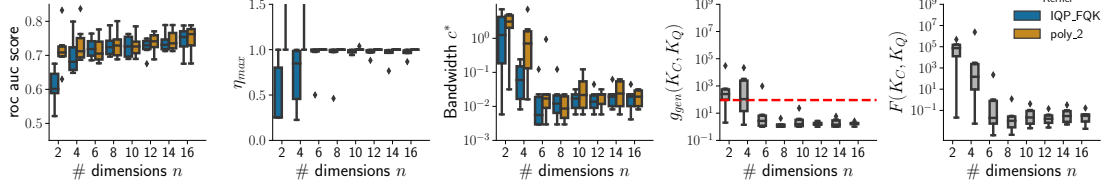
The quantum circuits used are shown in Fig. A.1 and Fig. A.2.

For the eigenvalue calculations, we diagonalize the matrix K normalized with the number of data points, i.e. $\text{eigen}(\frac{K}{N})$ as done in Ref. [14, 55].

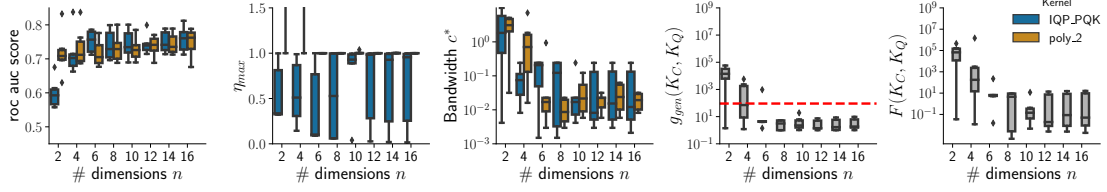


A.5. Supporting figures for bandwidth-tuned kernels

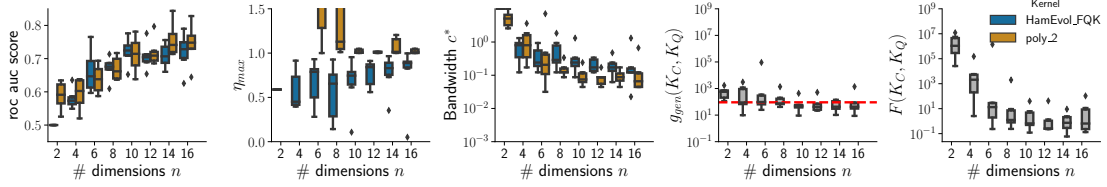
Similar to section 4.1.2, we reproduced the literature results (Fig. A.3) using FQK and PQK approaches based on the IQP encoding circuits with the `kMNIST28` dataset and FQKs based on the HamEvol encoding circuit for the `plasticc` dataset. Similarly, we also show the behavior of the remaining models as a function of the bandwidth (Fig. A.4). Finally, we extend the eigenvalue and task-model alignment figures (Fig. 4.3 and Fig. 4.4) to cover most of the simulated dimensions and all the classical and quantum models (Fig. A.5/A.6 and Fig. A.7/A.8 for `plasticc`/`kMNIST28`).



(A) IQP FQK for the kMNIST28 dataset.



(B) IQP PQK for the kMNIST28 dataset.



(C) HamEvol FQK for the plasticc dataset.

Figure A.3.: Repetition of Fig. 4.2 for the remaining datasets. Box plots of important quantities as a function of dimensionality, blue/yellow corresponds to the quantum/classical model and gray corresponds to the geometric difference $g(K_C, K_Q)$ and the Frobenius distance $F(K_C, K_Q)$ measures which depend on kernel matrices of both models. The boxes are constructed from the models with optimal bandwidths c^* , yielding the highest roc auc scores for each seed. Figures from left to right depict: roc auc score, largest eigenvalue $\eta_{\max}$, optimal bandwidth c^* , geometrical difference $g(K_C, K_Q)$ and Frobenius distance $F(K_C, K_Q)$. From top to bottom different models: (A) IQP FQK (kMNIST28), (B) IQP PQK (kMNIST28) and (C) HamEvol FQK (plasticc). The red dashed line for the geometrical difference corresponds to $\sqrt{N}$, the limit where no quantum advantage is possible. For increasing dimensions, the geometric difference and the Frobenius distance between the classical and quantum models decreases. This is reflected in similar performance and largest eigenvalue.

A. Appendix

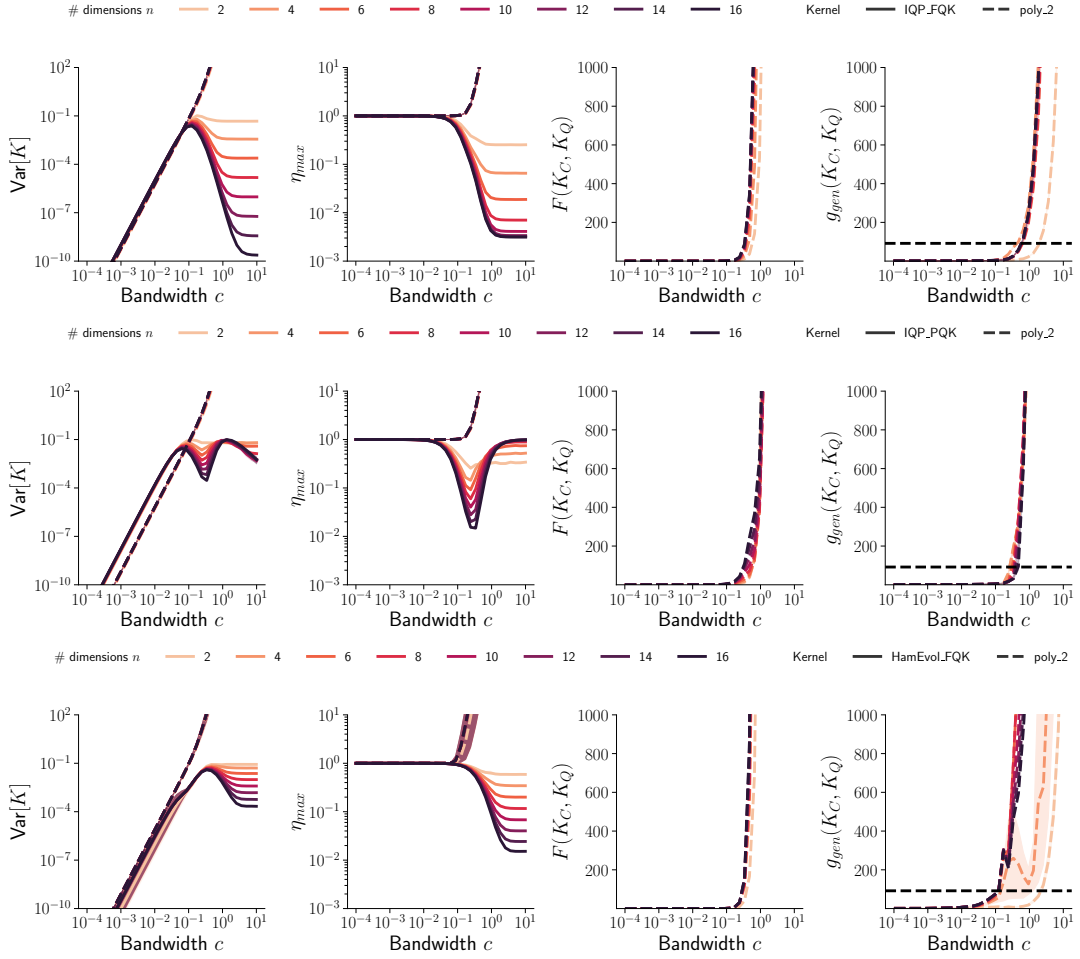


Figure A.4.: Repetition of 4.5 for the remaining datasets. Relevant measures as a function of bandwidth averaged over the different simulated seeds. Different colors correspond to different dimensionality, the darker the color the larger the number of dimensions. The shaded area corresponds to the standard deviation of the models. Dashed lines represent the classical models (here `poly_2`), whereas solid lines represent the quantum models. From top to bottom: (A) IQK (FQK) and kMNIST28, (B) IQK (PQK) and kMNIST28, (C) HamEvol (FQK) and plasticc. From left to right: variance of the kernel entries $\text{Var}[K]$, largest eigenvalue $\eta_{\max}$, Frobenius distance $F(K_C, K_Q)$ and geometrical difference $g(K_C, K_Q)$. The horizontal dot-dashed line in the geometrical difference plots corresponds to $\sqrt{N}$, the limit where no quantum advantage is possible.

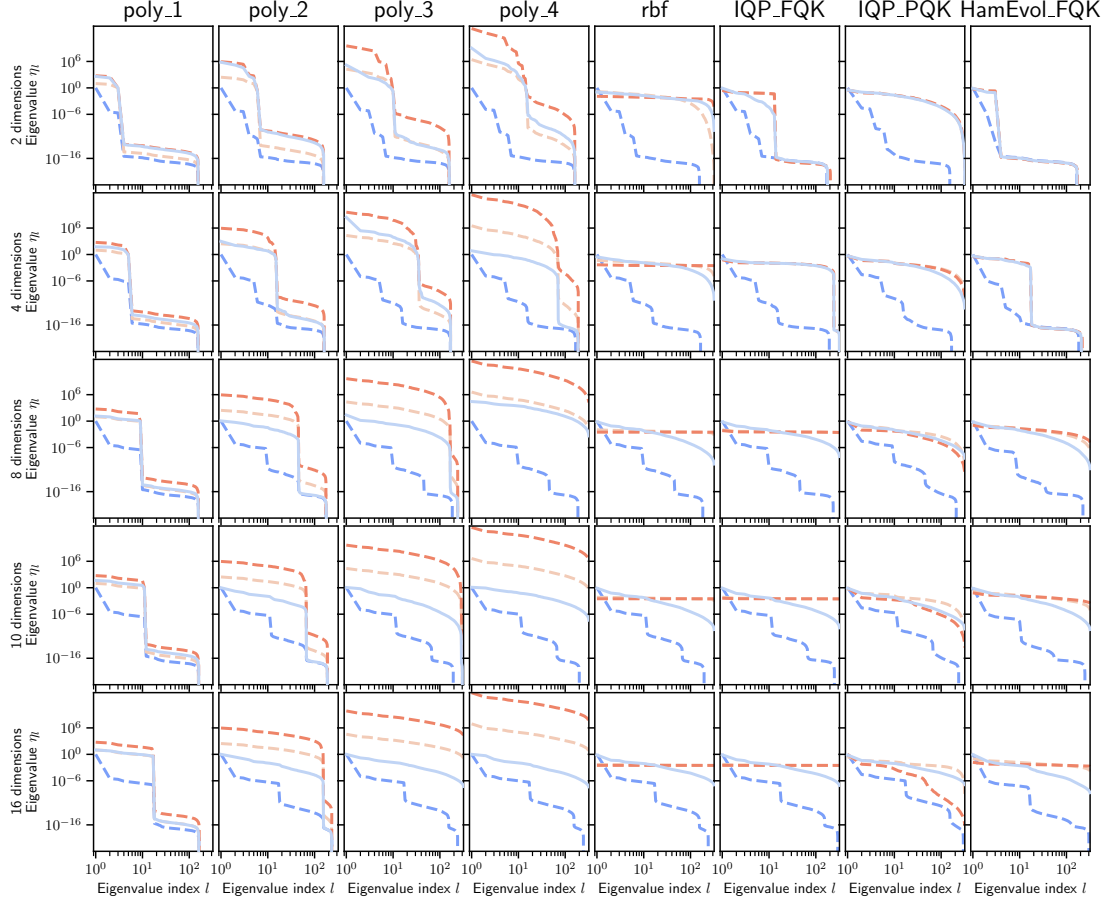


Figure A.5.: Eigenvalue spectra for increasing dimensionality classical and quantum models used for the plasticc experiments. Extension of Fig. 4.3 for the remaining kernels. From left to right, different kernel models: poly_1, poly_2, poly_3, poly_4, RBF, IQP FQK, IQP PQK and HamEvol FQK. From top to bottom, different dimensions: 2, 4, 8, 10, 16.

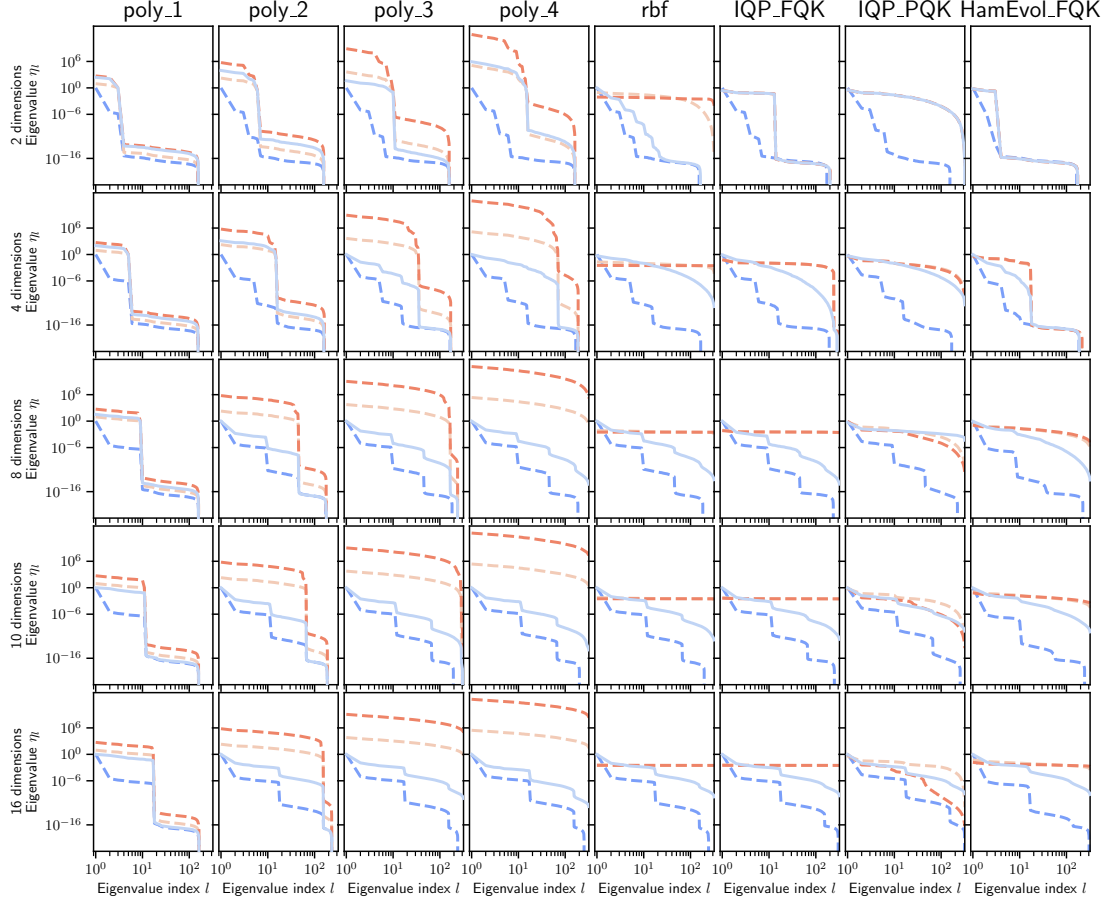


Figure A.6.: Eigenvalue spectra for increasing dimensionality classical and quantum models used for the kMNIST28 experiments.. Extension of Fig. 4.3 for the remaining kernels. From left to right, different kernel models: poly_1, poly_2, poly_3, poly_4, RBF, IQP FQK, IQP PQK and HamEvol FQK. From top to bottom, different dimensions: 2, 4, 8, 10, 16.

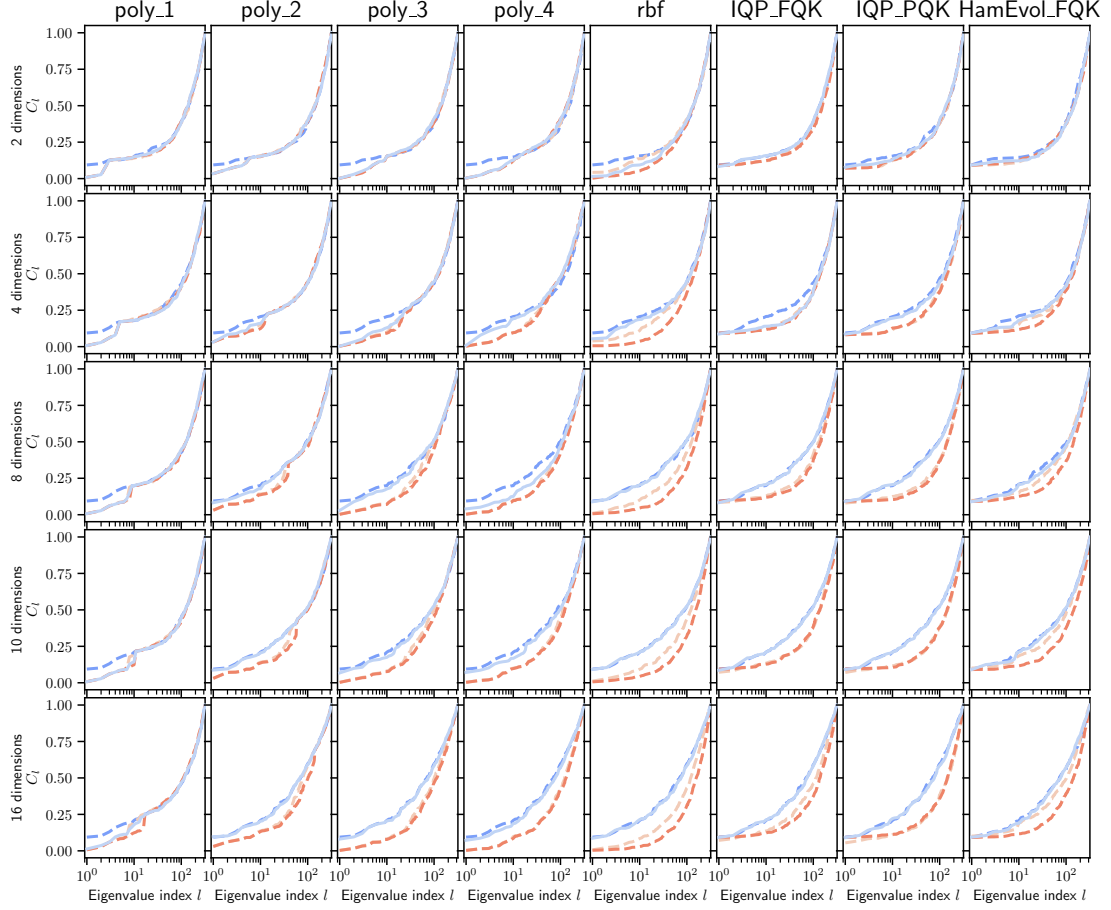


Figure A.7.: Task-model alignment for increasing dimensionality classical and quantum models used for the plasticc experiments. Extension of Fig. 4.4 for the remaining kernels. From left to right, different kernel models: poly_1, poly_2, poly_3, poly_4, RBF, IQP FQK, IQP PQK and HamEvol FQK. From top to bottom, different dimensions: 2, 4, 8, 10, 16.

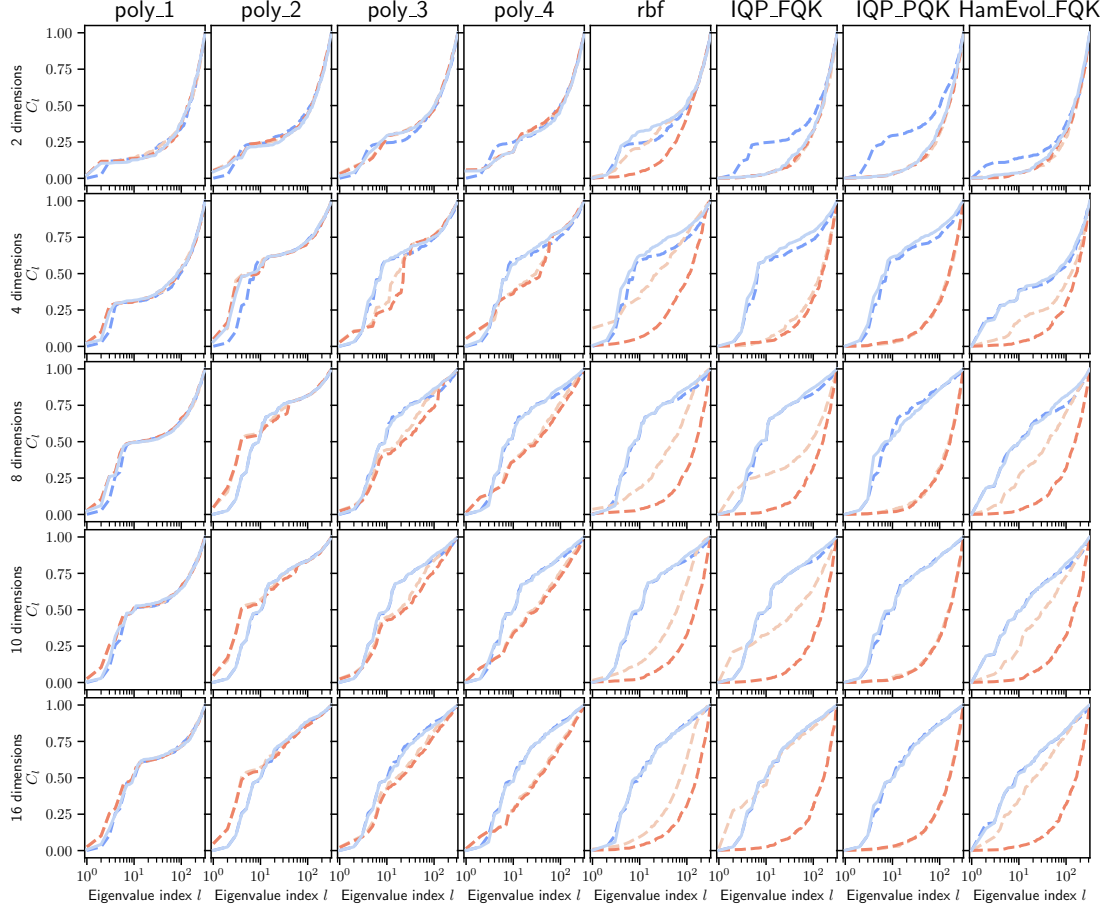


Figure A.8.: Task-model alignment for increasing dimensionality classical and quantum models used for the kMNIST28 experiments. Extension of Fig. 4.4 for the remaining kernels. From left to right, different kernel models: poly_1, poly_2, poly_3, poly_4, RBF, IQP FQK, IQP PQK and HamEvol FQK. From top to bottom, different dimensions: 2, 4, 8, 10, 16.

A.6. Separable R_X encoding circuit

Consider $\mathbf{x} = (x_1, \dots, x_n) \in \mathbb{R}^N$, the separable x rotation circuit is defined as,

$$U(\mathbf{x}) = \bigotimes_{j=1}^n U(x_j) = \bigotimes_{j=1}^n R_x(x_j) = \bigotimes_{j=1}^n e^{ix_j \sigma_x / 2} = \bigotimes_{j=1}^n \begin{pmatrix} \cos(x_j/2) & i \sin(x_j/2) \\ i \sin(x_j/2) & \cos(x_j/2) \end{pmatrix} \quad (\text{A.10})$$

Including L re-uploading layers and a quantum bandwidth factor c , that rescales the data, the feature map is given by,

$$\begin{aligned} \prod_l U(c\mathbf{x}) &= \bigotimes_{j=1}^n \prod_l U(cx_j) = \bigotimes_{j=1}^n \underbrace{U(cx_j)U(cx_j) \dots U(cx_j)}_{L \text{ times}} \\ &= \bigotimes_{j=1}^n U(cx_j)^L \\ &= \bigotimes_{j=1}^n R_x(cx_j)^L = \bigotimes_{j=1}^n e^{iLcx_j \sigma_x / 2} \\ &= \bigotimes_{j=1}^n R_x(Lcx_j) = \bigotimes_{j=1}^n U(Lcx_j). \end{aligned} \quad (\text{A.11})$$

The density matrix is given by,

$$\begin{aligned} \rho(\mathbf{x}) &= \left(\prod_l U(c\mathbf{x}) \right) |0\rangle \langle 0| \left(\prod_l U(c\mathbf{x}) \right)^\dagger = \bigotimes_{j=1}^n \rho(cx_j) \\ &= \bigotimes_{j=1}^n \begin{pmatrix} \cos^2(Lcx_j/2) & -i \cos(Lcx_j/2) \sin(cLx_j/2) \\ i \cos(Lcx_j/2) \sin(cx_j/2) & \sin^2(cLx_j/2) \end{pmatrix} \end{aligned} \quad (\text{A.12})$$

Then, the fidelity quantum kernel $k^{FQ}(\mathbf{x}, \mathbf{x}') = \text{tr}[\rho(c\mathbf{x})\rho(c\mathbf{y})]$ is given by,

$$k^{FQ}(\mathbf{x}, \mathbf{y}, c, L) = \text{tr}[\rho(\mathbf{x})\rho(\mathbf{x}')] \quad (\text{A.13})$$

$$= \text{tr} \left[\bigotimes_{j=1}^n \rho(x_j) \bigotimes_{j=1}^n \rho(y_j) \right] \quad (\text{A.14})$$

$$= \prod_{j=1}^n \text{tr}[\rho(x_j)\rho(y_j)] \quad (\text{A.15})$$

$$= \prod_{j=1}^n \cos\left(c \frac{L}{2} (x_j - y_j)\right)^2 \quad (\text{A.16})$$

$$= \prod_{j=1}^n k^{FQ}(x_j, y_j, c, L) \quad (\text{A.17})$$

Where, we used that $\text{tr}(A \otimes B) = \text{tr}(A) \text{tr}(B)$ and $\text{tr}[\rho(x_j)\rho(y_j)] = \cos\left(\frac{L}{2}(x_j - y_j)\right)^2$. A similar derivation, can be done for the RBF Projected Quantum Kernel. Finding,

$$k^{PQ}(\mathbf{x}, \mathbf{y}, c) = e^{-2\gamma(n - \sum_k^n \cos(Lc(x_k - y_k)))} \quad (\text{A.18})$$

A.6.1. Analytical Variance of the Fidelity Quantum Kernel

Consider a training dataset $\mathcal{D}$ of N points $\mathbf{x}_i$, where $\mathbf{x}_i \in \mathcal{X} \subset \mathbb{R}^n$, and every $\mathbf{x}_i \in \mathcal{X}$ follows a probability distribution $p(\mathbf{x})$. The variance of a random variable $k(\mathbf{x}, \mathbf{y})$ in $\mathcal{X}$ is given by,

$$\text{Var}[k(\mathbf{x}, \mathbf{y})]_{\mathcal{X}} = \mathbb{E}_{\mathcal{X}} [k(\mathbf{x}, \mathbf{y})^2] - \mathbb{E}_{\mathcal{X}} [k(\mathbf{x}, \mathbf{y})]^2 \quad (\text{A.19})$$

For simplicity consider that $n = 1$, so that $\mathbf{x}_i = x_i$. Then, the expectation value $\mathbb{E}_{\mathcal{X}} [k(x, y)]$ is given by,

$$\mathbb{E}_{\mathcal{X}} [k(x, y)] = \int \int k(x, y) p(x) p(y) dx dy \quad (\text{A.20})$$

and the expectation value of $\mathbb{E}_{\mathcal{X}} [k^2(x, y)]$ is given by,

$$\mathbb{E}_{\mathcal{X}} [k^2(x, y)] = \int \int k^2(x, y) p(x) p(y) dx dy \quad (\text{A.21})$$

.

Note that for the kernel of Eq. A.13, assuming that the distribution for each coordinate are independent, these equations can be easily extended to the n -dimensional case. For the probabilities, we assume $p(\mathbf{x}) = \prod_{j=1}^n p(x_j)$ and for the kernel, $k(\mathbf{x}, \mathbf{y}) = \prod_j k(x_j, y_j)$. Then, the expectation value is given by,

$$\mathbb{E}_{\mathcal{X}} [k(\mathbf{x}, \mathbf{y})] = \int \int \prod_{j=1}^n k(x_j, y_j) p(x_j) p(y_j) dx_j dy_j \quad (\text{A.22})$$

We can simplify to,

$$\mathbb{E}_{\mathcal{X}} [k(\mathbf{x}, \mathbf{y})] = \prod_{j=1}^n \int \int k(x_j, y_j) p(x_j) p(y_j) dx_j dy_j = \prod_{j=1}^n \mathbb{E} [k(x_j, y_j)] \quad (\text{A.23})$$

In Fig. A.9, we report the analytical distributions used for the derivations.

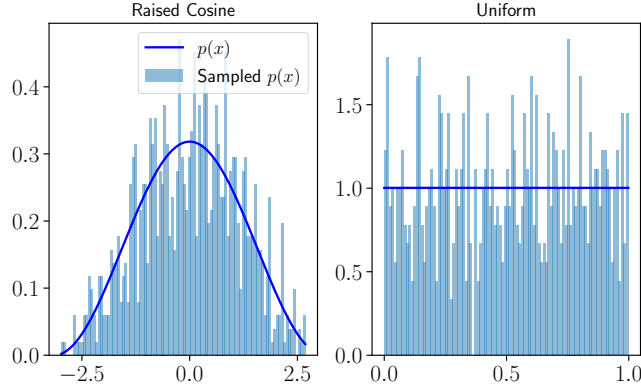


Figure A.9.: Raised cosine $p(x) = 1/2\pi (1 + \cos(x))$ and uniform distributions $p(x) = 1$ for $n = 1$. As used for the derivation of Eq. A.27 and Eq. A.36. For the numerical experiments, 1200 points were sampled.

Uniform distribution

Taking a uniform distribution for $x \in [0, 1]$, we have, $p(x) = 1$. Then, the expectation value of $k(x, y)$ is given by,

$$\mathbb{E}_{\mathcal{X}} [k(x, y)] = \int \int \cos^2 \left(\frac{Lc(x-y)}{2} \right) dx dy \quad (\text{A.24})$$

Solving with SymPy, we find that,

$$\mathbb{E}_{\mathcal{X}} [k(x, y)] = \frac{1}{2} - \frac{\cos(c)}{c^2} + \frac{1}{c^2} \quad (\text{A.25})$$

and the expectation value of $k^2(X, Y)$ is given by,

$$\mathbb{E}_{\mathcal{X}} [k^2(x, y)] = \int \int \cos^4 \left(\frac{Lc(x-y)}{2} \right) dx dy \quad (\text{A.26})$$

Solving with SymPy, we find that,

$$\mathbb{E}_{\mathcal{X}} [k^2(x, y)] = \frac{3}{8} - \frac{\cos(c)}{c^2} - \frac{1}{16} \frac{\cos(2c)}{c^2} + \frac{17}{16c^2}$$

Thus,

$$\text{Var}[k(x, y)] = \mathbb{E}_{\mathcal{X}} [k^2(x, y)] - \mathbb{E}_{\mathcal{X}} [k(x, y)]^2 \quad (\text{A.27})$$

$$= \frac{3}{8} - \frac{\cos(Lc)}{(Lc)^2} - \frac{1}{16} \frac{\cos(2Lc)}{(Lc)^2} + \frac{17}{16(Lc)^2} - \left(\frac{1}{2} - \frac{\cos(Lc)}{(Lc)^2} + \frac{1}{(Lc)^2} \right)^2 \quad (\text{A.28})$$

$$= \frac{1}{(Lc)^4} \left(\frac{1}{8}(Lc)^4 + \frac{1}{8}(Lc)^2 \sin^2(Lc) + \sin^2(Lc) + 2 \cos(Lc) - 2 \right) \quad (\text{A.29})$$

To extend to n -dimensions considering independent random distributions for each component, we apply Eq. A.23. Then, the variance of the fidelity quantum kernel is given by,

$$\text{Var}[k^{FQ}(\mathbf{x}, \mathbf{y}, c, L)] = \left(\mathbb{E} [k^2(x, y)] \right)^n - \left(\mathbb{E} [k(x, y)]^2 \right)^n. \quad (\text{A.30})$$

In the small bandwidth limit, by taking a fourth-order Taylor expansion, we find,

$$\text{Var}[k^{FQ}(\mathbf{x}, \mathbf{y}, c, L)] \approx \alpha(n)(Lc)^4, \text{ with } \alpha(n) = 0.000243n \quad (\text{A.31})$$

For the large bandwidth limit, we find that the variance is given by,

$$\text{Var}[k(x, y)]_{\mathcal{X}} \approx (a^n - b^n)/c^n, \text{ with } a = 3 \cdot 2^2, b = 2^3, c = 2^5. \quad (\text{A.32})$$

Cosine distribution

The cosine distribution for $x \in [-\pi, \pi]$ is given by,

$$p(x) = \frac{1}{2\pi} (1 + \cos(x)) \quad (\text{A.33})$$

which is an approximation of a Gaussian distribution. To find the analytical variance, we performing the same calculation as before,

$$\mathbb{E}_{\mathcal{X}} [k(x, y)] = \int \int \cos^2 \left(\frac{Lc(x-y)}{2} \right) p(x)p(y) dx dy \quad (\text{A.34})$$

$$= \frac{1}{2} + \frac{1}{2} \frac{\sin^2(\pi Lc)}{\pi^2 (Lc)^2 ((Lc)^2 - 1)^2} \quad (\text{A.35})$$

and

$$\begin{aligned}
\mathbb{E}_{\mathcal{X}}[k^2(x, y)] &= \int \int \cos^4\left(\frac{Lc(x-y)}{2}\right) p(x)p(y) dx dy \\
&= \frac{16(1-4(Lc)^2)^2 \sin^2(\pi(Lc))}{32\pi^2(4(Lc)^5 - 5(Lc)^3 + (Lc))^2} \\
&\quad + \frac{((Lc)^2 - 1)^2 (12\pi^2(Lc)^2(1-4(Lc)^2)^2 + \sin^2(2\pi(Lc)))}{32\pi^2(4(Lc)^5 - 5(Lc)^3 + (Lc))^2}.
\end{aligned}$$

Thus,

$$\text{Var}[k(x, y)] = \mathbb{E}_{\mathcal{X}}[k^2(x, y)] - \mathbb{E}_{\mathcal{X}}[k(x, y)]^2 \quad (\text{A.36})$$

$$= \frac{16(1-4(Lc)^2)^2 \sin^2(\pi(Lc))}{32\pi^2(4(Lc)^5 - 5(Lc)^3 + (Lc))^2} \quad (\text{A.37})$$

$$+ \frac{((Lc)^2 - 1)^2 (12\pi^2(Lc)^2(1-4(Lc)^2)^2 + \sin^2(2\pi(Lc)))}{32\pi^2(4(Lc)^5 - 5(Lc)^3 + (Lc))^2} \quad (\text{A.38})$$

$$- \left(\frac{1}{2} + \frac{1}{2} \frac{\sin^2(\pi Lc)}{\pi^2(Lc)^2((Lc)^2 - 1)^2} \right)^2. \quad (\text{A.39})$$

For n independent variables, we may use again Eq A.30. In the small bandwidth limit, by taking a fourth-order Taylor expansion, we find,

$$\text{Var}[k^{FQ}(\mathbf{x}, \mathbf{y}, c, L)] \approx \alpha(n)(Lc)^4, \text{ with } \alpha(n) = 0.7083n \quad (\text{A.40})$$

For the large bandwidth limit, we find that the variance is given by,

$$\text{Var}[k(x, y)]_{\mathcal{X}} \approx (a^n - b^n)/d^n, \text{ with } a = 3 \cdot 2^2, b = 2^3, d = 2^5. \quad (\text{A.41})$$

A.7. Implementation of Quantum Differential Equation Solvers

This thesis makes two contributions to the QML library sQUlearn [17]. On one hand we implemented a differential equation solver based on QNNs and on the other the building blocks for a differential equation solver based on QKs.

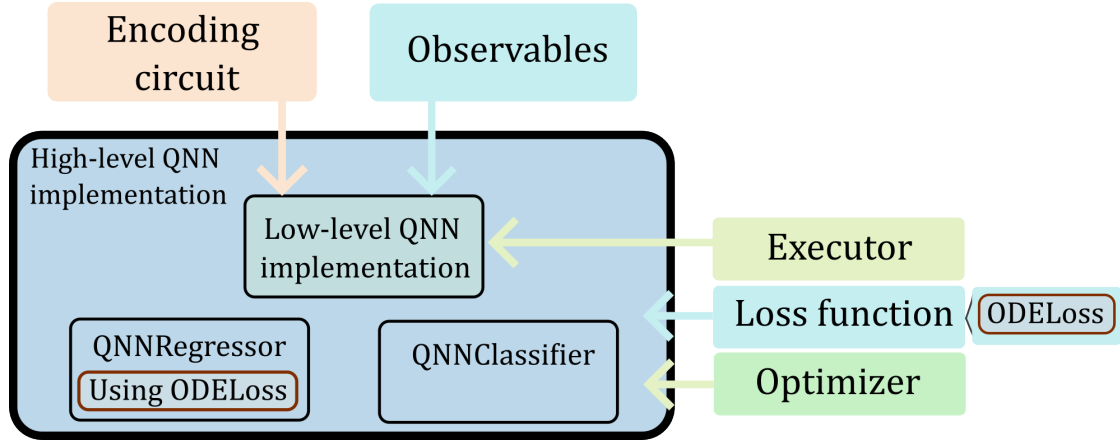


Figure A.10.: Overview of the main components of the sQUlearn QNN engine including the differential equation solver contained in ODELoss. Adapted from Ref. [17]

The technical details of these implementation are described below.

A.7.1. QNN Differential Equation Solver

The QNN engine of sQUlearn is capable of minimizing arbitrary loss functions. The main components of the sQUlearn QNN engine are shown in Fig. A.10. Our implementation¹ consists of creating a new loss class in the loss module, that translates a given DE (Eq. 2.23) to a loss function (Eq. 2.25). This can then be minimized by the QNN engine. Therefore, the user simply needs to provide an arbitrary DE in the form of Eq. 2.23. In Fig. A.11 we show an example of how to use the QNN engine to solve a differential equation.

Under the hood, besides the translation of the DE to a loss function, we also implemented the gradient of the differential equation loss function. This is a necessary requirement for certain optimization algorithms, such as the Adam optimizer [81]. We report the gradient of the loss function in Eq. A.44.

A.7.2. QK Differential Equation Solver

Implementing the differential equation solver for the QK engine is somewhat analogous to the previously discussed QNN engine. Nevertheless, the current implementation

¹see <https://github.com/sQUlearn/squlearn/pull/278>

of the QK engine is in ongoing renovation². Our local implementation, consists of a new loss class in the loss module. The loss class minimizes the linear coefficients of an ansatz constructed from the kernel (Eq. 2.53). The loss function depends on the derivatives of the kernel function $\frac{\partial K(x,y)}{\partial x}$. For FQKs they are straightforward to calculate and for PQKs with gaussian outer kernel and XYZ measurements (Eq. 2.39) they are given by Eq. A.47 and Eq. A.48 of the Appendix³. Similar to the QNN implementation, the user provides a given DE in the form of Eq. 2.23, which is converted to a loss function of the form of Eq. 2.25 that can then be minimized by sQUlearn.

²see <https://github.com/sQUlearn/sqlearn/issues/274>

³For the sQUlearn implementation, see <https://github.com/sQUlearn/sqlearn/pull/277>

```

1 import numpy as np
2 import sympy as sp
3 from sqlearn import Executor
4 from sqlearn.encoding_circuit import KyriienkoEncodingCircuit
5 from sqlearn.observables import SummedPaulis
6 from sqlearn.qnn import QNNRegressor, ODELoss, get_lr_decay
7 from sqlearn.optimizers import Adam
8
9 nqubits = 6
10 circuit = KyriienkoEncodingCircuit(
11     num_qubits=nqubits,
12     encoding_style="chebyshev_tower",
13     variational_arrangement="HEA",
14     num_features=1,
15     num_encoding_layers=1,
16     num_variational_layers=5,
17 )
18 observable = SummedPaulis(nqubits, include_identity=True)
19 x, f, dfdx, dfdx dx = sp.symbols("x f dfdx dfdx dx")
20 k, lamb = 0.1, 20
21 eq = lamb * sp.exp(-lamb * x * k) * sp.sin(lamb * x) + lamb * k * f + dfdx
22 initial_values = [1]
23 loss_ODE_sp = ODELoss(
24     eq,
25     symbols_involved_in_ODE=[x, f, dfdx],
26     initial_values=initial_values,
27     boundary_handling="pinned",
28 )
29 ode_regressor = QNNRegressor(
30     circuit,
31     observable,
32     Executor("pennylane", shots=None),
33     loss_ODE_sp,
34     Adam(options={"maxiter": 125, "tol": 0.00009, "lr": get_lr_decay(0.05, 0.02, 125)}),
35     circuit.generate_initial_parameters(seed=1),
36     observable.generate_initial_parameters(seed=1),
37     opt_param_op=True,
38 )
39 #We train the model to solve the ODE
40 x_space = np.linspace(0, 0.9, 15).reshape(-1, 1)
41 ref_values = np.zeros_like(x_space)
42 ode_regressor.fit(x_space, ref_values)
43 y = ode_regressor.predict(x)

```

Figure A.11.: Example of the QNN engine to solve the differential equation, $F\left(\frac{df}{dx}, f, x\right) = \lambda + e^{-\lambda k x} \sin(\lambda x) + \lambda k f + \frac{df}{dx} = 0$ with $f(0) = 1$. The chebyshev tower encoding style proposed by Kyriienko et al. [15] (Eq. 15), and the HEA variational arrangement (section IIIB of Ref. [15]) were used.

A.8. Gradient of the loss of a DE

Some popular NN optimizers (e.g Adam [81]) require the gradient of the loss function to minimize it. In the main text, we defined the loss function of a DE as in Eq. 2.47 to be used by a QNN engine,

$$\mathcal{L}_\theta[\ddot{f}, \dot{f}, f, x] = \sum_j^N L[F[\ddot{f}_\theta, \dot{f}_\theta, f_\theta, x]_j] + \eta(L(f_\theta(0), \dot{u}_0) + L(f_\theta(0), \dot{u}_0)) \quad (\text{A.42})$$

$$= \sum_j^N (F[\ddot{f}_\theta, \dot{f}_\theta, f_\theta, x]_j)^2 + \eta(f_\theta(0) - u_0)^2 + \eta(\dot{f}_\theta(0) - \dot{u}_0)^2. \quad (\text{A.43})$$

The gradient of this loss is given by,

$$\begin{aligned} \frac{\partial \mathcal{L}_\theta}{\partial \theta_i} &= \sum_j^N 2(F[\ddot{f}_\theta, \dot{f}_\theta, f_\theta, x]_j) \frac{\partial}{\partial \theta_i} (F[\ddot{f}_\theta, \dot{f}_\theta, f_\theta, x]_j) \\ &\quad + 2\eta(f_\theta(0) - u_0) \left. \frac{\partial f_\theta(x)}{\partial \theta_i} \right|_{x=0} + 2\eta(\dot{f}_\theta(0) - \dot{u}_0) \left. \frac{\partial \dot{f}_\theta(x)}{\partial \theta_i} \right|_{x=0} \\ &= \sum_j^N 2(F[\ddot{f}_\theta, \dot{f}_\theta, f_\theta, x]_j) \left(\frac{\partial F_j}{\partial \ddot{f}_\theta} \frac{\partial \ddot{f}_\theta}{\partial \theta_i} + \frac{\partial F_j}{\partial \dot{f}_\theta} \frac{\partial \dot{f}_\theta}{\partial \theta_i} + \frac{\partial F_j}{\partial f_\theta} \frac{\partial f_\theta}{\partial \theta_i} \right) \\ &\quad + 2\eta(f_\theta(0) - u_0) \left. \frac{\partial f_\theta(x)}{\partial \theta_i} \right|_{x=0} + 2\eta(\dot{f}_\theta(0) - \dot{u}_0) \left. \frac{\partial \dot{f}_\theta(x)}{\partial \theta_i} \right|_{x=0}. \end{aligned} \quad (\text{A.44})$$

A.9. Derivatives of RBF PQKs

A.9.1. First derivative

Let us consider a 1-dimensional problem, $x \in \mathbb{R}^1$. A PQK can be seen as:

$$k_{\text{PQK}}(x, y) = g(\mathbf{O}(x), \mathbf{O}(y)) \quad (\text{A.45})$$

where g is the envelope function and $\mathbf{O} \in \mathbb{R}^{L \times 1}$ the vector that contains the expectation values of the L observables of the PQK. The derivative of the PQK with respect to the input x is:

$$\frac{\partial k_{\text{PQK}}(x, y)}{\partial x} = \frac{\partial k_{\text{PQK}}(g(\mathbf{O}(x), \mathbf{O}(y)))}{\partial x} = \frac{\partial k_{\text{PQK}}}{\partial g} \frac{\partial g}{\partial \mathbf{O}} \frac{\partial \mathbf{O}}{\partial x} = \frac{\partial g}{\partial \mathbf{O}} \frac{\partial \mathbf{O}}{\partial x} \quad (\text{A.46})$$

For N sample points, i.e, $x \in \mathbb{R}^{N \times 1}$ and $y \in \mathbb{R}^{N \times 1}$, in index notation the derivative of the PQK with respect to the input x is:

$$\left(\frac{\partial K_{\text{PQK}}}{\partial x} \right)_{nj} = \sum_l \left(\frac{\partial g(\mathbf{O}(x = x_n), \mathbf{O}(y = y_j))}{\partial \mathbf{O}(x = x_n)} \right)_l \left(\frac{\partial \mathbf{O}(x_n)}{\partial x} \right)_l \quad (\text{A.47})$$

$$\left(\frac{\partial K_{\text{PQK}}}{\partial x} \right)_{nj} = \sum_l \left(\frac{\partial g}{\partial \mathbf{O}} \right)_{(njl)} \left(\frac{\partial \mathbf{O}}{\partial x} \right)_{(nl)} \quad (\text{A.48})$$

For M features, i.e, $x \in \mathbb{R}^{N \times M}$, the derivative of the PQK with respect to the input x is:

$$\left(\frac{\partial K_{\text{PQK}}}{\partial x} \right)_{njm} = \sum_l \left(\frac{\partial g(\mathbf{O}(x = x_n), \mathbf{O}(y = y_j))}{\partial \mathbf{O}(x = x_n)} \right)_l \left(\frac{\partial \mathbf{O}(x_n)}{\partial x} \right)_{lm} \quad (\text{A.49})$$

A.9.2. Second derivative

From Eq. A.46, the second derivative of the PQK with respect to the input x is:

$$\frac{\partial^2 k_{\text{PQK}}(x, y)}{\partial x^2} = \frac{\partial}{\partial x} \left(\frac{\partial g(\mathbf{O}(x), \mathbf{O}(y))}{\partial \mathbf{O}(x)} \frac{\partial \mathbf{O}(x)}{\partial x} \right) \quad (\text{A.50})$$

$$= \frac{\partial}{\partial x} \left(\frac{\partial g(\mathbf{O}(x), \mathbf{O}(y))}{\partial \mathbf{O}(x)} \right) \frac{\partial \mathbf{O}}{\partial x} + \frac{\partial g(\mathbf{O}(x), \mathbf{O}(y))}{\partial \mathbf{O}(x)} \frac{\partial}{\partial x} \left(\frac{\partial \mathbf{O}(x)}{\partial x} \right) \quad (\text{A.51})$$

$$= \sum_{l,m} \frac{\partial g}{\partial O_l O_m} \frac{\partial O_l}{\partial x} \frac{\partial O_m}{\partial x} + \frac{\partial g}{\partial \mathbf{O}} \frac{\partial^2 \mathbf{O}}{\partial x^2} \quad (\text{A.52})$$

$$= \sum_l \frac{\partial^2 g}{\partial O_l^2} \left(\frac{\partial O_l}{\partial x} \right)^2 + \sum_{l \neq m} \frac{\partial g}{\partial O_l O_m} \frac{\partial O_l}{\partial x} \frac{\partial O_m}{\partial x} + \frac{\partial g}{\partial \mathbf{O}} \frac{\partial^2 \mathbf{O}}{\partial x^2} \quad (\text{A.53})$$

$$= \sum_l \frac{\partial^2 g}{\partial O_l^2} \left(\frac{\partial O_l}{\partial x} \right)^2 + 2 \sum_{l \neq m, l < m} \frac{\partial g}{\partial O_l O_m} \frac{\partial O_l}{\partial x} \frac{\partial O_m}{\partial x} + \frac{\partial g}{\partial \mathbf{O}} \frac{\partial^2 \mathbf{O}}{\partial x^2} \quad (\text{A.54})$$

$$(\text{A.55})$$

Acknowledgments

Many people and institutions were responsible for providing me with the financial, emotional, and practical support to pursue my master's studies in Germany. I would like to thank them all.

I would like to thank Prof. Jörg Main for agreeing to be the official advisor of this thesis and giving me the opportunity to participate in the social activities of the ITP1. I would also like to thank the members of the ITP1 for always being very receptive to me in both social and academic events.

Similarly, I want to thank Prof. Mathias Scheurer for agreeing to be the second examiner of the thesis. Furthermore, I want to thank him for accepting me to undertake a small research project in the context of the artificial intelligence specialization at the university. For this project, I would also like to thank João Sobral, who provided me with technical support and was very patient with me.

I would like to thank my advisor Dr. Jan Schnabel for his help during my work. I am very grateful for the academic freedom to discuss and research new ideas. I am grateful for all the technical help and input. I always felt that an excellent working environment was created and that my ideas and doubts were always very welcomed and encouraged. Moreover, I am especially grateful for the many inputs and suggestions on both professional topics not only limited to the master's thesis.

I would also like to thank the Quantum Computing group at the Fraunhofer IPA. I would like to thank Dr. Marco Roth for the opportunity to work in the group and for the feedback given throughout my time at the Fraunhofer. Similarly, I would also like to thank Dr. David Kreplin for his feedback and support throughout my research, particularly for his suggestions on differential equations solvers and sQUlearn. Also, I would like to thank the rest of the group Frederic Rapp, Moritz Willmann, and Manuel Hagelüken, for being very welcoming with me and sympathetic with me. Furthermore, I would like to thank Timo Leitz for the technical support on the workstation for the numerical simulations.

I would like to thank the Max Planck Society for the financial support through the IMPRS-CMS scholarship during the first three semesters of my studies in Germany and the initial support given upon my arrival. On a similar note, I would like to thank the Fraunhofer Institute for the financial and administrative support during the development of this work. Furthermore, I want to thank Germany's Bundesministerium für Bildung und Forschung and the Carl-Zeiss-Stiftung for their financial support and recognition through the Deutschland Stipendium.

I am very grateful to the University of Stuttgart for the excellent lectures and professors during my master's program. Additionally, I would like to thank the Studierendenwerk Stuttgart and its entire staff for their support with housing, assistance, and the Mensa.

I would also like to thank the organizational staff of the master's program, Patric Rommel and Mia Kumric.

I would like to extend my thanks to my master cohort friends, Sebastian, Levon, Abhi, Matthew, Reuel, Xuankai, Ana, and Irene. I am especially grateful to Reuel and Xuankai for the many enjoyable hours and days we spent learning, studying, and discussing physics together.

I would like to thank all of my non-academics friends for their help and trust. In particular, I would like to thank my friends in Stuttgart. I am grateful to my friend Emilia for the confidence and fun times we have shared since our arrival in Stuttgart. I would like to thank Oriol for our friendship and all the time spent doing sports, discussing science, and sharing our daily lives. I am grateful to my friends Dori and Fabian for providing me with a strong support network and for being my first local friends, especially for always being there to help me navigate cultural and social situations. Moreover, I would like to thank my friend Nicholas for his constant friendship and his great help with improving my German skills. Also, I would like to thank my friends Marco and Henrique for the important friendship during the initial months in Stuttgart.

I would like to thank my partner, Milena, for everything: for her patience and contagious happiness over these past years, and for being a source of motivation and support.

Finally, I want to thank my entire family, as I would not be here without them. Especially, I am very grateful to my parents, Elvira and Julio, for the care, love, and financial support that I have received throughout my life. I would like to thank my siblings, Diego and Julia, for always being close to me. Last but not least, I am deeply grateful for the care and unconditional love from my aunt Isa and my grandparents Juan and Rosalba.